

**Green, Shirelle**

---

**From:** Unknown@Unknown.com  
**Sent:** Wednesday, March 10, 2004 3:21 PM  
**To:** STIC-EIC2100  
**Subject:** Generic form response

*ResponseHeader=Commercial Database Search Request*

*AccessDB#= 116637*

*LogNumber= SA*

*Searcher=*

*SearcherPhone=*

*SearcherBranch=*

*MyDate=Wed Mar 10 15:19:26 EST 2004*

*submitto=STIC-EIC2100@uspto.gov*

*Name=Derek Rutten*

*Empno=79877*

*Phone=703-605-5233*

*Artunit=2122*

*Office=CPK2 5B46*

*Serialnum=09/810,191*

*PatClass=717/122,145*

*Earliest=10/25/2000*

*Format3=email*

*Searchtopic=When computer program source code is compiled, the compiler translates the source from a human readable format to a machine readable format called object code which is*



# STIC Search Report

## EIC 2100

STIC Database Tracking Number: 116504

TO: *Dr. Rutter*

Location: *5B46*

Art Unit: *2122*

Case Serial Number: *09/810,551*

From: Carol Wong

Location: EIC 2100

PK2-4B33

Phone: 305-9729

[carol.wong@uspto.gov](mailto:carol.wong@uspto.gov)

### Search Notes

Dear Examiner *Rutter*:

Attached are the search results (from commercial databases) for your case. Two formats are provided: hard copy and floppy disk. Pls be aware that bolding of search terms is lost in the electronic version.

Color tags on the hard copy results mark the patents/articles which appear to be most relevant to the case. Pls review all documents, since untagged items might also be of interest. If you wish to order the complete text of any document, pls submit request(s) directly to the EIC2100 Reference Staff located in PK2-4B40.

Pls call if you have any questions or suggestions for additional terminology, or a different approach to searching the case. Finally, pls complete the attached Search Results Feedback Form, as the EIC/STIC is continually soliciting examiners' opinion of the search service.

Thanks,  
Carol



stored in a separate file. Object code is often divided into sections containing the actual machine program instructions (text), as well as variable information (data). Once a program is compiled into object code, it is difficult or impossible to reconstruct the original source file. The invention embeds the source code directly in the object code file as a separate section. When the original source file is further edited, the compiler compares the edited source file with the original source file stored in the object code, and only compiles those portions that are changed. The new compiled code section then replaces the old in the object file, and the source section of the object file is also updated to reflect the change. This is an effort to reduce compilation time. The key feature is that the source code is being stored in the object file.

Terms: compiler, object file format, object code, source code,

Comments=I'm usually in the office M-F 6:30-3

send=SEND

File 256:SoftBase:Reviews,Companies&Prods. 82-2004/Feb  
(c)2004 Info.Sources Inc  
File 2:INSPEC 1969-2004/Mar W1  
(c) 2004 Institution of Electrical Engineers  
File 6:NTIS 1964-2004/Mar W1  
(c) 2004 NTIS, Intl Cpyrght All Rights Res  
File 8:Ei Compendex(R) 1970-2004/Mar W1  
(c) 2004 Elsevier Eng. Info. Inc.  
File 34:SciSearch(R) Cited Ref Sci 1990-2004/Mar W1  
(c) 2004 Inst for Sci Info  
File 35:Dissertation Abs Online 1861-2004/Feb  
(c) 2004 ProQuest Info&Learning  
File 65:Inside Conferences 1993-2004/Mar W2  
(c) 2004 BLDSC all rts. reserv.  
File 94:JICST-EPlus 1985-2004/Mar W1  
(c)2004 Japan Science and Tech Corp(JST)  
File 95:TEME-Technology & Management 1989-2004/Feb W4  
(c) 2004 FIZ TECHNIK  
File 99:Wilson Appl. Sci & Tech Abs 1983-2004/Feb  
(c) 2004 The HW Wilson Co.  
File 111:TGG Natl.Newspaper Index(SM) 1979-2004/Mar 16  
(c) 2004 The Gale Group  
File 144:Pascal 1973-2004/Mar W1  
(c) 2004 INIST/CNRS  
File 202:Info. Sci. & Tech. Abs. 1966-2004/Feb 27  
(c) 2004 EBSCO Publishing  
File 233:Internet & Personal Comp. Abs. 1981-2003/Sep  
(c) 2003 EBSCO Pub.  
File 266:FEDRIP 2004/Jan  
Comp & dist by NTIS, Intl Copyright All Rights Res  
File 434:SciSearch(R) Cited Ref Sci 1974-1989/Dec  
(c) 1998 Inst for Sci Info  
File 483:Newspaper Abs Daily 1986-2004/Mar 12  
(c) 2004 ProQuest Info&Learning  
File 583:Gale Group Globalbase(TM) 1986-2002/Dec 13  
(c) 2002 The Gale Group  
File 603:Newspaper Abstracts 1984-1988  
(c)2001 ProQuest Info&Learning

Set	Items	Description
S1	106	SOURCECOD?
S2	1394037	SOURCE OR HUMANREAD? OR HUMAN(1W)READ????
S3	26517	S2(1W){CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCEDURE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? OR F- ILE OR FILES OR SUBFILE? ?}
S4	6799	S2(1W){PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? OR - LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?}
S5	431	S2(2N)COMMENT?
S6	1902	S2(2N)OBJECT? ?
S7	1874202	OBJECT OR MACHINEREAD? OR MACHINE OR MACHINELANGUAGE? OR N- ATIVE OR NONHUMAN OR NON(1W)HUMAN
S8	1359	POSITION? ?(1W){RELOCAT? OR INDEPENDENT}
S9	14361	S7:S8(2W){CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCED- URE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? ? OR FILE OR FILES OR SUBFILE? ?}
S10	96873	S7:S8(2W){PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? - OR LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?}
S11	9	S7:S8(2W)SUBPROGRAM?
S12	1675	BYTECOD???? ? OR BYTE(1W){CODE? ? OR CODING? ?}
S13	0	S2(1W)SUBPROGRAM?
S14	380	(S1 OR S3:S6 OR S13)(3N){STORE? ? OR STORED OR STORING OR - STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?}
S15	2488	(S1 OR S3:S6 OR S13)(3N){INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?}
S16	2	OBJECTCOD? OR OBJECTFIL?
S17	5162	S2(2N){REMARK? ? OR ANNOTATION? OR EXPLANATION? OR OBSERVA-

TION? OR NOTES OR FOOTNOTE?)

S18 9 S17(3N)(STORE? ? OR STORED OR STORING OR STORAGE OR EMBED?  
OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)

S19 58 S17(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR  
ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)

S20 173 (S14:S15 OR S18:S19) AND (S9:S12 OR S16)

S21 33 (S14:S15 OR S18:S19)(10N)(S9:S12 OR S16)

S22 27 S21 NOT OBJECT(1W)ORIENT?? ?(1W)PROGRAM?

S23 6 S21 NOT S22

S24 2819 (S1 OR S3:S6 OR S13 OR S17)(5N)(COMPIL??? ? OR TRANSLAT? OR  
CONVERT? OR CONVERSION? OR INTERPRET? OR ASSEMBL? OR MACROAS-  
SEMBL?)

S25 26 S20 AND S24

S26 51 S21 OR S25

S27 3 S26/2001:2004

S28 48 S26 NOT S27

S29 43 RD (unique items)

29/7/10 (Item 3 from file: 2)

DIALOG(R)File 2:INSPEC

(c) 2004 Institution of Electrical Engineers. All rts. reserv.

5621431 INSPEC Abstract Number: C9708-6150E-007

**Title: The design of distributed hyperlinked programming documentation**

Author(s): Friendly, L.

Author Affiliation: Sun Microsyst. Inc., Palo Alto, CA, USA

Conference Title: Hypermedia Design. Proceedings of the International  
Workshop on Hypermedia Design (IWHD'95) p.151-62

Editor(s): Fraisse, S.; Garzotto, F.; Isakowitz, T.; Nanard, J.; Nanard,  
M.

Publisher: Springer-Verlag, Berlin, Germany

Publication Date: 1996 Country of Publication: Germany xiii+252 pp.

ISBN: 3 540 19985 3 Material Identity Number: XX95-01430

Conference Title: Proceedings of International Workshop on Hypermedia  
Design

Conference Sponsor: Univ. Montpellier; District de Montpellier

Conference Date: 1-2 June 1995 Conference Location: Montpellier,  
France

Language: English Document Type: Conference Paper (PA)

Treatment: Practical (P)

Abstract: HotJava is a World-Wide Web browser that adds dynamic behavior  
to hypertext access by supporting the downloading and execution of  
architecture-neutral, interactive applets from inside a Web page. HotJava  
is written in Java, a new **object-oriented language** and environment  
developed at Sun Microsystems. The paper describes the design of the  
documentation for Java's application programming interface (API), for  
display and distribution on the World-Wide Web. Following in the footsteps  
of the literate programming paradigm, the documentation was automatically  
generated from source code. The author designed a syntax for documentation  
comments which are **embedded** in the **source code** and parsed by the Java  
**compiler** to produce HTML markup. The display environment of the  
World-Wide-Web presented many challenging design requirements for  
readability, usability, and navigation. The paper discusses the design  
process, from augmenting the source code commenting syntax to designing the  
layout for the Web pages. The resulting product is a set of integrated Web  
pages which are hyperlinked, highly readable, and easily navigated. The API  
documentation was first published on the World-Wide Web in December of  
1994. (13 Refs)

Subfile: C

Copyright 1997, IEE

29/7/13 (Item 6 from file: 2)

DIALOG(R)File 2:INSPEC

(c) 2004 Institution of Electrical Engineers. All rts. reserv.

5411097 INSPEC Abstract Number: C9612-6140D-021

**Title: Handling source code and images in natOO: The concept and some**

## applications

Author(s): Dahm, M.

Author Affiliation: Lehrstuhl fur Messtechnik, Tech. Hochschule Aachen, Germany

Conference Title: EuroForth '94 p.133-40

Publisher: Microprocessor Eng, Southampton, UK

Publication Date: 1994 Country of Publication: UK ix+158 pp.

Material Identity Number: XX95-00553

Conference Title: Proceedings The European Forth Conference. EuroForth '94

Conference Date: 4-6 Nov. 1994 Conference Location: Winchester, UK

Availability: MicroProcessor Engineering Ltd., 133 Hill Lane, Southampton, SO15 5AF, UK

Language: English Document Type: Conference Paper (PA)

Treatment: Practical (P)

Abstract: At the Lehrstuhl fur Messtechnik of RWTH Aachen University an object-oriented FORTH (OOF) with extensions for programming in a natural language style (natOOF) has been developed, which has been introduced and described in former conferences. This paper presents a way of organising source code and saved system states (images) of natOOF. One major advantage of natOOF is its late binding which, in addition to other aspects, makes it possible to redefine functions (methods) without having to redefine all dependent methods. This feature is best supported when the user is not bound to the traditional source handling using various source files in various directories himself but can rely on automatic source code retrieval and storage. For this purpose, a project-structure is introduced. Each object (e.g. a variable, a class or a method) is defined as part of a project. The file and the position of an object's source is stored coded in its header. Thus, an object's code can be retrieved and edited without any knowledge of the filename nor the position within the file. This concept is elaborated to include handling of logfiles which contain every change made to an image during a session. Care was taken to ensure that multiple users can work on a problem without interfering with each other's code. The projects form the background of an integrated development system of editors, workspaces and applications. (11 Refs)

Subfile: C

Copyright 1996, IEE

29/7/22 (Item 15 from file: 2)

DIALOG(R)File 2:INSPEC

(c) 2004 Institution of Electrical Engineers. All rts. reserv.

02270776 INSPEC Abstract Number: C84030698

Title: A convenient 6502 assembler in BASIC

Author(s): Gutekunst, T.

Journal: Mikro- und Kleincomputer vol.5, no.6 p.61-74

Publication Date: Dec. 1983 Country of Publication: Switzerland

CODEN: MKLED2 ISSN: 0251-0006

Language: German Document Type: Journal Paper (JP)

Treatment: Practical (P)

Abstract: An assembler written for the CBM 4032 and working with a CBM 2031 floppy disc and a CBM 4022P printer is described. It consists of two parts. The actual assembler program translates in two passes the source code (stored on a discette) into the object code and returns this as object file to the discette. Hard copy can be produced if desired from the printer. The assembly program is about 8500 bytes in length and requires more than 4 kbytes storage space for variables. The loader program reads the object code again from the discette and stores it in the computer. The mode of operation of the assembler is explained with the aid of a program example termed 'DEMO'. Because the assembler program is written in BASIC, the translation of the source program takes a relatively long time, and this constitutes a slight disadvantage. (0 Refs)

Subfile: C

?t29/7/24-25,31,36

29/7/24 (Item 17 from file: 2)

DIALOG(R)File 2:INSPEC

(c) 2004 Institution of Electrical Engineers. All rts. reserv.

01737823 INSPEC Abstract Number: B81042103, C81029711

**Title: A controlled environment for the development of switching system software**

Author(s): Lovitt, G.J.; Hills, M.T.L.; Hurst, R.S.

Conference Title: IEE Fourth International Conference on Software Engineering for Telecommunication Switching Systems p.64-7

Publisher: IEE, London, UK

Publication Date: 1981 Country of Publication: UK vi+220 pp.

Conference Date: 20-24 July 1981 Conference Location: Coventry, UK

Language: English Document Type: Conference Paper (PA)

Treatment: Practical (P)

Abstract: ITT's system, known as the Software Development Support System (SDSS) and operating on IBM 370 computers, contains two principal components. The first is a set of tools for **translating source code into object code**, for linking **together** separately **compiled** modules of code and for building systems. The second, known as the Software Production Monitor System (SPMS), provides a unified controlling interface between the user, the tool set and the data base of software items. (0 Refs)

Subfile: B C

29/7/25 (Item 18 from file: 2)

DIALOG(R)File 2:INSPEC

(c) 2004 Institution of Electrical Engineers. All rts. reserv.

01671342 INSPEC Abstract Number: C81014799

**Title: The Master Catalog System for CP/M users**

Author(s): Hallen, R.

Author Affiliation: State Dept., Accra, Washington, DC, USA

Journal: Kilobaud Microcomputing no.12 p.188-90

Publication Date: Dec. 1980 Country of Publication: USA

ISSN: 0192-4575

Language: English Document Type: Journal Paper (JP)

Treatment: Economic aspects (E)

Abstract: The system diskette contains 18 programs which **include** both **source** and **object codes**. Three **programs** list disk directories alphabetically, with program size in K, and total vacant space available on the disk. These listings are three or four columns wide, depending upon the width of the screen. The system comprises a series of programs used to create and display the catalog. (0 Refs)

Subfile: C

29/7/31 (Item 4 from file: 6)

DIALOG(R)File 6:NTIS

(c) 2004 NTIS, Intl Cpyrght All Rights Res. All rts. reserv.

0378961 NTIS Accession Number: AD-759 348/XAB

**A Library Management Program for the 813 Disk File**

(Final rept)

Toothman, H. L.

Naval Research Lab Washington D C

Corp. Source Codes: 251950

Report No.: NRL-MR-2570; NRL-COMPUTER BULL-31

Mar 73 46p

Journal Announcement: GRAI7312

Order this product from NTIS by: phone at 1-800-553-NTIS (U.S. customers); (703)605-6000 (other countries); fax at (703)321-8547; and email at orders@ntis.fedworld.gov. NTIS is located at 5285 Port Royal Road, Springfield, VA, 22161, USA.

NTIS Prices: PC A03/MF A01

Contract No.: NRL-D01-03-A

RANDISK is a CDC 3800 FORTRAN and assembly language program which allows the **storage** of data, and **source** and **object language files** on the 813 disk file by a user assigned name. These files can be recalled by name

and transferred to a logical unit for further use. The user has some control of logical units under RANDISK and may document his library on tape rapidly. Data compression of card image files can be performed. The source language listing is included. (Author Modified Abstract)

29/7/36 (Item 2 from file: 202)  
DIALOG(R) File 202:Info. Sci. & Tech. Abs.  
(c) 2004 EBSCO Publishing. All rts. reserv.

2002240

**Electronic dictionary (Patent).**

Author(s): Kobayashi, S; Yoshida, T.

Patent Number(s): US 4481607

Publication Date: Nov 6, 1984

Language: English

Place of Publication: United States

Document Type: Patent

Record Type: Abstract

Journal Announcement: 2000

An electronic dictionary which translates a word from a source language to an object language by key control and displays the word on a display section, comprising: key input means including at least one forward key and at least one reverse key, and a translation key; memory means for storing words of the source language in an order of word arrangement corresponding to that of a dictionary, and for storing words of the object language, wherein the stored words have associated addresses; word search means coupled to said key input means and to said memory means for specifying the addresses of the words of the source language stored in said memory means, and for enabling the display of the words of the source language on the display section, wherein said word search means specifies said addresses sequentially in a forward order of the stored word arrangement in response to operation of said forward key, and specifies said addresses sequentially in a reverse order of the stored word arrangement in response to operation of said reverse key; and translation means coupled to said key input means and to said memory means for selecting from said memory means that word of the object language which corresponds to the translation of the word of the source language in response to operation to said translation key wherein said translation means is operative to specify the address of the word of the object language, and for enabling the display of the word of the object language the address of which is specified, on the display section.



File 9:Business & Industry(R) Jul/1994-2004/Mar 12  
(c) 2004 Resp. DB Svcs.  
File 16:Gale Group PROMT(R) 1990-2004/Mar 15  
(c) 2004 The Gale Group  
File 47:Gale Group Magazine DB(TM) 1959-2004/Mar 15  
(c) 2004 The Gale group  
File 148:Gale Group Trade & Industry DB 1976-2004/Mar 09  
(c)2004 The Gale Group  
File 160:Gale Group PROMT(R) 1972-1989  
(c) 1999 The Gale Group  
File 275:Gale Group Computer DB(TM) 1983-2004/Mar 15  
(c) 2004 The Gale Group  
File 570:Gale Group MARS(R) 1984-2004/Mar 15  
(c) 2004 The Gale Group  
File 621:Gale Group New Prod.Annou.(R) 1985-2004/Mar 15  
(c) 2004 The Gale Group  
File 636:Gale Group Newsletter DB(TM) 1987-2004/Mar 15  
(c) 2004 The Gale Group  
File 649:Gale Group Newswire ASAP(TM) 2004/Mar 12  
(c) 2004 The Gale Group

Set	Items	Description
S1	136	SOURCECOD?
S2	2832497	SOURCE OR HUMANREAD? OR HUMAN(1W)READ????
S3	93925	S2(1W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCEDURE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? OR F- ILE OR FILES OR SUBFILE? ?)
S4	9145	S2(1W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? OR - LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S5	6057	S2(2N)COMMENT?
S6	3505	S2(2N)OBJECT? ?
S7	1666816	OBJECT OR MACHINEREAD? OR MACHINE OR MACHINELANGUAGE? OR N- ATIVE OR NONHUMAN OR NON(1W)HUMAN
S8	1272	POSITION? ?(1W)(RELOCAT? OR INDEPENDENT)
S9	29930	S7:S8(2W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCED- URE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? ? OR FILE OR FILES OR SUBFILE? ?)
S10	57415	S7:S8(2W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? - OR LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S11	9	S7:S8(2W)SUBPROGRAM?
S12	3728	BYTECOD???? ? OR BYTE(1W)(CODE? ? OR CODING? ?)
S13	1	S2(1W)SUBPROGRAM?
S14	1519	(S1 OR S3:S6 OR S13)(3N)(STORE? ? OR STORED OR STORING OR - STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S15	10781	(S1 OR S3:S6 OR S13)(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S16	16	OBJECTCOD? OR OBJECTFIL?
S17	4926	S2(2N)(REMARK? ? OR ANNOTATION? OR EXPLANATION? OR OBSERVA- TION? OR NOTES OR FOOTNOTE?)
S18	28	S17(3N)(STORE? ? OR STORED OR STORING OR STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S19	139	S17(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S20	1647	(S14:S15 OR S18:S19) AND (S9:S12 OR S16)
S21	282	(S14:S15 OR S18:S19)(10N)(S9:S12 OR S16)
S22	252	S21 NOT OBJECT(1W)ORIENT?? ?(1W)PROGRAM?
S23	30	S21 NOT S22
S24	8542	(S1 OR S3:S6 OR S13 OR S17)(5N)(COMPIL? OR TRANSLAT? OR CO- NVERT? OR CONVERSION? OR INTERPRET? OR ASSEMBL? OR MACROASSEM- BL?)
S25	50	S21(S)S24
S26	77	S23 OR S25
S27	10	S26/2001:2004
S28	67	S26 NOT S27
S29	45	RD (unique items)

29/3,K/19 (Item 4 from file: 148)  
DIALOG(R)File 148:Gale Group Trade & Industry DB  
(c)2004 The Gale Group. All rts. reserv.

05918764 SUPPLIER NUMBER: 12509832 (USE FORMAT 7 OR 9 FOR FULL TEXT)  
**Programming the Macintosh with Think C 5.0.**  
MacPhail, John  
Library Software Review, v11, n2, p13(5)  
March-April, 1992  
DOCUMENT TYPE: evaluation ISSN: 0742-5759 LANGUAGE: ENGLISH  
RECORD TYPE: FULLTEXT  
WORD COUNT: 3710 LINE COUNT: 00290

... Symantec programmers for developing their products.  
The Integrated THINK C Environment  
The THINK C application integrates file management, a **source code** editor, a **compiler**, and an **object code** linker. Besides simplifying the programmer's job, this integration is more efficient because one hand...

...the way THINK C deals with its project documents. The project document comprises all the **compiled code** of all the separate **source files**, along with references to the source files so that THINK C can find the source...

29/3,K/24 (Item 3 from file: 160)  
DIALOG(R)File 160:Gale Group PROMT(R)  
(c) 1999 The Gale Group. All rts. reserv.

01843398  
**risC Product Provides C Programmers with C-like Assembler**  
News Release October 31, 1987 p. 1

... C while maintaining the precision and compactness of assembly language programming. risC lends itself to **Object Oriented Programming** structures. It contains a complete object-oriented messaging kernal ( **source code included** ) which allows risC objects (.EXE files) to pass messages back and forth. The powerful language...

29/3,K/29 (Item 4 from file: 275)  
DIALOG(R)File 275:Gale Group Computer DB(TM)  
(c) 2004 The Gale Group. All rts. reserv.

01706299 SUPPLIER NUMBER: 16271196 (USE FORMAT 7 OR 9 FOR FULL TEXT)  
**Implementing code reuse. (object-based features in CA-Clipper) (Tutorial)**  
Gutierrez, Dan D.  
Data Based Advisor, v12, n10, p154(4)  
Oct, 1994  
DOCUMENT TYPE: Tutorial ISSN: 0740-5200 LANGUAGE: ENGLISH  
RECORD TYPE: FULLTEXT; ABSTRACT  
WORD COUNT: 1747 LINE COUNT: 00170

... Enter the VO repository, an object-oriented archive that maintains information about each application entity, including the **source code**, the **compiled object code**, the module in which the entity resides, and all dependencies between entities throughout the application...  
?t29/3,k/39-40

29/3,K/39 (Item 14 from file: 275)  
DIALOG(R)File 275:Gale Group Computer DB(TM)  
(c) 2004 The Gale Group. All rts. reserv.

01252776 SUPPLIER NUMBER: 06833221 (USE FORMAT 7 OR 9 FOR FULL TEXT)  
**EBASIC compiler. (languages)**  
Deutschman, Jay

DG Review, v9, n1, p36(2)

July, 1988

ISSN: 1050-9127

LANGUAGE: ENGLISH

RECORD TYPE: FULLTEXT; ABSTRACT

WORD COUNT: 1238

LINE COUNT: 00099

... INCLUDE segments, variable references and line numbers.

Commenting and More

Since EBASIC is a true **compiler**, programmers can **include** **comments** in their **source programs** without effecting the size of their **object programs**. In addition to the traditional REM statement, programmers can include remarks at the end of...

29/3,K/40 (Item 15 from file: 275)

DIALOG(R)File 275:Gale Group Computer DB(TM)

(c) 2004 The Gale Group. All rts. reserv.

01245663 SUPPLIER NUMBER: 07002113 (USE FORMAT 7 OR 9 FOR FULL TEXT)

**AXOS 3.3. (Software Review) (Advanced Indexed Organization System) (Product Watch) (evaluation)**

Wright, Victor E.

PC Tech Journal, v6, n10, p136(2)

Oct, 1988

DOCUMENT TYPE: evaluation

ISSN: 0738-0194

LANGUAGE: ENGLISH

RECORD TYPE: FULLTEXT; ABSTRACT

WORD COUNT: 1163

LINE COUNT: 00092

... CPU registers can call AXOS's 57 file-management functions.

Three intermediate processors (furnished in **assembly** -language **source code** and **object - file form**) **together** with a generic AXOS programming language interface (in a 3,584-byte library file) ease...

?

File 696:DIALOG Telecom. Newsletters 1995-2004/Mar 15  
(c) 2004 The Dialog Corp.  
File 9:Business & Industry(R) Jul/1994-2004/Mar 15  
(c) 2004 Resp. DB Svcs.  
File 15:ABI/Inform(R) 1971-2004/Mar 15  
(c) 2004 ProQuest Info&Learning  
File 98:General Sci Abs/Full-Text 1984-2004/Feb  
(c) 2004 The HW Wilson Co.  
File 484:Periodical Abs Plustext 1986-2004/Mar W1  
(c) 2004 ProQuest  
File 813:PR Newswire 1987-1999/Apr 30  
(c) 1999 PR Newswire Association Inc  
File 613:PR Newswire 1999-2004/Mar 15  
(c) 2004 PR Newswire Association Inc  
File 635:Business Dateline(R) 1985-2004/Mar 13  
(c) 2004 ProQuest Info&Learning  
File 810:Business Wire 1986-1999/Feb 28  
(c) 1999 Business Wire  
File 610:Business Wire 1999-2004/Mar 15  
(c) 2004 Business Wire.  
File 369:New Scientist 1994-2004/Mar W1  
(c) 2004 Reed Business Information Ltd.  
File 370:Science 1996-1999/Jul W3  
(c) 1999 AAAS  
File 20:Dialog Global Reporter 1997-2004/Mar 16  
(c) 2004 The Dialog Corp.  
File 624:McGraw-Hill Publications 1985-2004/Mar 15  
(c) 2004 McGraw-Hill Co. Inc  
File 634:San Jose Mercury Jun 1985-2004/Mar 15  
(c) 2004 San Jose Mercury News  
File 647:CMP Computer Fulltext 1988-2004/Mar W1  
(c) 2004 CMP Media, LLC  
File 674:Computer News Fulltext 1989-2004/Mar W1  
(c) 2004 IDG Communications

Set	Items	Description
S1	196	SOURCECOD?
S2	5179471	SOURCE OR HUMANREAD? OR HUMAN(1W)READ????
S3	44042	S2(1W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCEDURE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? OR F- ILE OR FILES OR SUBFILE? ?)
S4	5455	S2(1W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? OR - LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S5	6884	S2(2N)COMMENT?
S6	1836	S2(2N)OBJECT? ?
S7	1463616	OBJECT OR MACHINEREAD? OR MACHINE OR MACHINELANGUAGE? OR N- ATIVE OR NONHUMAN OR NON(1W)HUMAN
S8	1191	POSITION? ?(1W)(RELOCAT? OR INDEPENDENT)
S9	11798	S7:S8(2W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCED- URE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? ? OR FILE OR FILES OR SUBFILE? ?)
S10	36244	S7:S8(2W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? - OR LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S11	2	S7:S8(2W)SUBPROGRAM?
S12	1739	BYTECOD???? ? OR BYTE(1W)(CODE? ? OR CODING? ?)
S13	0	S2(1W)SUBPROGRAM?
S14	674	(S1 OR S3:S6 OR S13)(3N)(STORE? ? OR STORED OR STORING OR - STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S15	4472	(S1 OR S3:S6 OR S13)(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S16	6	OBJECTCOD? OR OBJECTFIL?
S17	5169	S2(2N)(REMARK? ? OR ANNOTATION? OR EXPLANATION? OR OBSERVA- TION? OR NOTES OR FOOTNOTE?)
S18	10	S17(3N)(STORE? ? OR STORED OR STORING OR STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)

S19 141 S17(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR  
ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)  
S20 537 (S14:S15 OR S18:S19) AND (S9:S12 OR S16)  
S21 101 (S14:S15 OR S18:S19)(10N)(S9:S12 OR S16)  
S22 26 S21/2001:2004  
S23 75 S21 NOT S22  
S24 64 RD (unique items)

24/3,K/13 (Item 10 from file: 15)  
DIALOG(R)File 15:ABI/Inform(R)  
(c) 2004 ProQuest Info&Learning. All rts. reserv.

00595799 92-10972  
C Languages: Oceans Apart  
Forer, Douglas  
InfoWorld v14n5 PP: 55-68 Feb 3, 1992  
ISSN: 0199-6649 JRNL CODE: IFW  
WORD COUNT: 8231

...TEXT: To measure project build time, we forced the compiler to recompile  
all the necessary program **source files**. We **included** the time to link  
the resulting **object** and resource **files** into a final executable file.  
When the DOS command line compilers and linkers were used...  
?t24/3,k/61

24/3,K/61 (Item 6 from file: 647)  
DIALOG(R)File 647:CMP Computer Fulltext  
(c) 2004 CMP Media, LLC. All rts. reserv.

00629798 CMP ACCESSION NUMBER: EET19890410S2584  
C++ becomes more popular as users apply it to system software development:  
C programmers push for object-orientation (Technology update)  
RAY WEISS  
ELECTRONIC ENGINEERING TIMES, 1989, n 533, 61  
PUBLICATION DATE: 890410  
JOURNAL CODE: EET LANGUAGE: English  
RECORD TYPE: Fulltext  
SECTION HEADING: TECHNOLOGY UPDATE  
WORD COUNT: 4815

... source code, which can be compiled by standard C compiler. Schwarz  
claims that the C **source code**, which **embeds object-oriented**  
**programming** constructs, can be deployed to Unix workstation for target  
application execution.

Objective-C

C++ is...the Glockenspiel C++ translator and includes incremental  
compilation, automatic dependency control, incremental linking and an  
**integrated source / object code** debugger. "We leave room in the code  
space for expansion, emphasizing interactive speed." Goldstine says...  
?

File 347:JAPIO Nov 1976-2003/Nov(Updated 040308)

(c) 2004 JPO & JAPIO

File 350:Derwent WPIX 1963-2004/UD,UM &UP=200416

(c) 2004 THOMSON DERWENT

File 348:EUROPEAN PATENTS 1978-2004/Mar W01

(c) 2004 European Patent Office

File 349:PCT FULLTEXT 1979-2002/UB=20040304,UT=20040226

(c) 2004 WIPO/Univentio

Set	Items	Description
S1	7	AU='MITSUMORI M'
S2	1	AU='MITSUMORI MASATO C O HITACHI LTD'
S3	2	AU='ASAKI S':AU='ASAKI S I'
S4	2	AU='HOSOTANI H'
S5	1	AU='HOSOTANI HIROYUKI C O SOFTWARE ENG CO LTD'
S6	2	S1:S2 AND S3:S5
S7	24	AU='ASAKA S'
S8	16	AU='ASAKA SHINJI':AU='ASAKA SHINJI C O TEIJIN SEIKI CO LTD'
S9	2	S1:S2 AND S7:S8
S10	0	S9 NOT S6

6/9/1 (Item 1 from file: 350)  
DIALOG(R)File 350:Derwent WPIX  
(c) 2004 THOMSON DERWENT. All rts. reserv.

014625422 \*\*Image available\*\*  
WPI Acc No: 2002-446126/200248  
Related WPI Acc No: 2003-734495  
XRPX Acc No: N02-351506

Compile method involves generating object program by compiling source  
program, which is stored in object program file along with source program  
Patent Assignee: HITACHI LTD (HITA ); HITACHI SOFTWARE ENG CO LTD (HISF )  
Inventor: ASAKA S; HOSOTANI H; MITSUMORI M  
Number of Countries: 026 Number of Patents: 001  
Patent Family:  
Patent No Kind Date Applicat No Kind Date Week  
EP 1202171 A2 20020502 EP 2001106817 A 20010319 200248 B

Priority Applications (No Type Date): JP 2000332110 A 20001025  
Patent Details:  
Patent No Kind Lan Pg Main IPC Filing Notes  
EP 1202171 A2 E 15 G06F-009/45  
Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT  
LI LT LU LV MC MK NL PT RO SE SI TR

Abstract (Basic): EP 1202171 A2  
NOVELTY - An object program generated by compiling a source  
program, is stored in an object program file (112) along with the  
source program. The source program is associated with the object  
program in the object program file.  
DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for:  
(1) Compiler  
(2) Recorded medium storing compile program.  
USE - Compile method.  
ADVANTAGE - Increases execution speed of compiled source program by  
optimization process. A procedure being compiled can be searched for  
any change made and compiled when necessary. A source program can be  
easily read or it can be encrypted to make it readable. Large  
proportion of compiling time is saved. Procedures can be updated  
easily.  
DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of  
compiler.  
Object program file (112)  
pp; 15 DwgNo 1/9  
Title Terms: COMPILE; METHOD; GENERATE; OBJECT; PROGRAM; COMPILE; SOURCE;  
PROGRAM; STORAGE; OBJECT; PROGRAM; FILE; SOURCE; PROGRAM  
Derwent Class: T01  
International Patent Class (Main): G06F-009/45  
File Segment: EPI  
Manual Codes (EPI/S-X): T01-F05A; T01-F07; T01-S03

6/5/2 (Item 1 from file: 348)  
DIALOG(R)File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

01423124  
Compile method and program recording medium  
Kompiliervverfahren und - Programmaufzeichnungsmedium  
Methode de compilation et media d'enregistrement de programme  
PATENT ASSIGNEE:

Hitachi, Ltd., (204145), 6 Kanda Surugadai 4-chome, Chiyoda-ku, Tokyo  
101-8010, (JP), (Applicant designated States: all)  
HITACHI SOFTWARE ENGINEERING CO., LTD., (678781), 81, Onoecho-6-chome  
Naka-ku, Yokohama, (JP), (Applicant designated States: all)  
INVENTOR:  
Mitsumori, Masato, c/o Hitachi, Ltd. , 5-1, Marunouchi 1-chome,  
Chiyoda-ku, Tokyo 100-8220, (JP)  
Asaka, Shinji, c/o Hitachi, Ltd., 5-1, Marunouchi 1-chome, Chiyoda-ku,  
Tokyo 100-8220, (JP)

\* Hosotani, Hiroyuki, c/o Software Eng., Co., Ltd. , 6-81, Onoe-cho,  
Naka-ku, Yokohama-shi, Kanagawa 231--0015, (JP

LEGAL REPRESENTATIVE:

Strehl Schubel-Hopf & Partner (100941), Maximilianstrasse 54, 80538  
Munchen, (DE)

PATENT (CC, No, Kind, Date): EP 1202171 A2 020502 (Basic)

APPLICATION (CC, No, Date): EP 2001106817 010319;

PRIORITY (CC, No, Date): JP 2000332110 001025

DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI;  
LU; MC; NL; PT; SE; TR

EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI

INTERNATIONAL PATENT CLASS: G06F-009/45

ABSTRACT EP 1202171 A2

A compiler (102), that generates an object program file (112) from a source program (101) in which a plurality of procedures are written, compiles procedures (func1, func2), by regarding the procedures as source-program compile units (602, 803, 804), to generate corresponding object-program compile units (601, 801, 802). A plurality of object-program compile units generated are output to a memory (106) together with the corresponding source-program compile units (602, 803, 804). When compiling a source program (101) in which one procedure has been changed, the compiler (102) compiles only the source-program compile unit corresponding to the changed procedure.

ABSTRACT WORD COUNT: 98

NOTE:

Figure number on first page: 1

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 020502 A2 Published application without search report  
LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	200218	1722
SPEC A	(English)	200218	3576
Total word count - document A			5298
Total word count - document B			0
Total word count - documents A + B			5298



File 347:JAPIO Nov 1976-2003/Nov(Updated 040308)

(c) 2004 JPO & JAPIO

File 350:Derwent WPIX 1963-2004/UD,UM &UP=200417

(c) 2004 Thomson Derwent

Set	Items	Description
S1	1	SOURCECOD?
S2	970857	SOURCE OR HUMANREAD? OR HUMAN(1W)READ????
S3	4539	S2(1W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCEDURE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? OR F- ILE OR FILES OR SUBFILE? ?)
S4	4839	S2(1W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? OR - LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S5	40	S2(2N)COMMENT?
S6	2574	S2(2N)OBJECT? ?
S7	1697110	OBJECT OR MACHINEREAD? OR MACHINE OR MACHINELANGUAGE? OR N- ATIVE OR NONHUMAN OR NON(1W)HUMAN
S8	1337	POSITION? ?(1W)(RELOCAT? OR INDEPENDENT)
S9	7306	S7:S8(2W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCED- URE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? ? OR FILE OR FILES OR SUBFILE? ?)
S10	8447	S7:S8(2W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? - OR LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S11	5	S7:S8(2W)SUBPROGRAM?
S12	621	BYTECOD???? ? OR BYTE(1W)(CODE? ? OR CODING? ?)
S13	2	S2(1W)SUBPROGRAM?
S14	1017	(S1 OR S3:S6 OR S13)(3N)(STORE? ? OR STORED OR STORING OR - STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S15	677	(S1 OR S3:S6 OR S13)(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S16	0	OBJECTCOD? OR OBJECTFIL?
S17	431	S2(2N)(REMARK? ? OR ANNOTATION? OR EXPLANATION? OR OBSERVA- TION? OR NOTES OR FOOTNOTE?)
S18	8	S17(3N)(STORE? ? OR STORED OR STORING OR STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S19	20	S17(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S20	316	(S14:S15 OR S18:S19) AND (S9:S12 OR S16)
S21	132	(S14:S15 OR S18:S19)(10N)(S9:S12 OR S16)
S22	131	S21 NOT OBJECT(1W)ORIENT?? ?(1W)PROGRAM?
S23	1	S21 NOT S22
S24	2701	(S1 OR S3:S6 OR S13 OR S17)(5N)(COMPIL??? ? OR TRANSLAT? OR CONVERT? OR CONVERSION? OR INTERPRET? OR ASSEMBL? OR MACROAS- SEMBL?)
S25	77	S21 AND S24
S26	8069	IC='G06F-009/45'
S27	5070	MC='T01-F05A'
S28	3890	MC='T01-F07'
S29	73	S21 AND S26:S28
S30	48	S25 AND S29
S31	69	S21 AND S26:S27
S32	7	S31 AND S28
S33	50	S30 OR S32
S34	50	IDPAT (sorted in duplicate/non-duplicate order)
S35	50	IDPAT (primary/non-duplicate records only)
S36	29	S25 NOT S35
S37	23	S29 NOT S35
S38	52	S36:S37
S39	52	IDPAT (sorted in duplicate/non-duplicate order)
S40	52	IDPAT (primary/non-duplicate records only)
?		

35/9/2 (Item 2 from file: 350)  
DIALOG(R) File 350:Derwent WPIX  
(c) 2004 Thomson Derwent. All rts. reserv.

015684169 \*\*Image available\*\*  
WPI Acc No: 2003-746358/200370  
XRPX Acc No: N03-598040

Program product for testing consistency of machine code and source files, has instructions executed to produce source file attribute record including file location information, file modified date, during compilation of file source code

Patent Assignee: FUJITSU LTD (FUIT )  
Inventor: KIYOTA K; MURAKAMI S  
Number of Countries: 002 Number of Patents: 002  
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 20030167423	A1	20030904	US 2003360737	A	20030210	200370 B
JP 2003256202	A	20030910	JP 200255963	A	20020301	200370

Priority Applications (No Type Date): JP 200255963 A 20020301

Patent Details:

Patent No Kind Lan Pg Main IPC Filing Notes

US 20030167423 A1 36 H02H-003/05

JP 2003256202 A 25 G06F-009/44

Abstract (Basic): US 20030167423 A1

NOVELTY - The program product has instructions executed to produce a source file attribute record (3b) including file location information (2a) and information that is to be updated while modifying the source file e.g. last modified date (2b), when a source code in a source file (1) is compiled into a machine code. The source file attribute record is then added to the machine code file.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

(1) method which supports consistency test of machine code file and source file;

(2) system which supports consistency test of machine code file and source file; and

(3) computer-readable medium storing computer program which supports consistency test of machine code file and source file.

USE - For use with computer system, for testing consistency of compiled machine code file with respect to original version and current version of its source file.

ADVANTAGE - It is easier for the user to check whether a compiled machine code program reflects all modifications made to its source files. Accuracy is ensured in testing the consistency of a large collection of executive files provided as library resources for software development.

DESCRIPTION OF DRAWING(S) - The figure explains the execution of consistency test program in computer system.

source file (1)  
file attributes (2,5)  
file location information (2a)  
last modified date (2b)  
machine code file (3)  
source file attribute record (3b)  
pp; 36 DwgNo 1/21

Title Terms: PROGRAM; PRODUCT; TEST; CONSISTENCY; MACHINE; CODE; SOURCE; FILE; INSTRUCTION; EXECUTE; PRODUCE; SOURCE; FILE; ATTRIBUTE; RECORD; FILE; LOCATE; INFORMATION; FILE; MODIFIED; DATE; COMPILE; FILE; SOURCE; CODE

Derwent Class: T01

International Patent Class (Main): G06F-009/44; H02H-003/05

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-J20C; T01-S03

35/9/3 (Item 3 from file: 350)  
DIALOG(R) File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

015418843 \*\*Image available\*\*

WPI Acc No: 2003-480983/200345

XRPX Acc No: N03-382480

Source code compilation method for just-in-time compiler, involves generating native machine code by updating each memory location storing references of source code, with value stored in associated register

Patent Assignee: LUEH G (LUEH-I)

Inventor: LUEH G

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 20030079211	A1	20030424	US 2001920274	A	20010731	200345 B

Priority Applications (No Type Date): US 2001920274 A 20010731

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
US 20030079211	A1	10	G06F-009/45	

Abstract (Basic): US 20030079211 A1

NOVELTY - A particular source code block including references to values stored in memory locations, is transformed into a block with references to values stored in associated registers. A compensation native machine code is generated to update each memory location with a value from the associated register so that a native machine code is provided.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

(1) just-in-time (JIT) compiler; and

(2) machine-readable medium storing source code compilation method.

USE - For just-in-time (JIT) compiler for compilation of source code program written in programming languages such as C++ and Java, into native machine code.

ADVANTAGE - Enables ensuring accurate exception reporting for the programming languages.

DESCRIPTION OF DRAWING(S) - The figure shows a flowchart illustrating source code compilation process.

pp; 10 DwgNo 2/5

Title Terms: SOURCE; CODE; COMPILE; METHOD; TIME; COMPILE; GENERATE; NATIVE ; MACHINE; CODE; UPDATE; MEMORY; LOCATE; STORAGE; REFERENCE; SOURCE; CODE ; VALUE; STORAGE; ASSOCIATE; REGISTER

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-F05G3; T01-J20B; T01-S01C; T01-S03

35/9/6 (Item 6 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

014625422 \*\*Image available\*\*

WPI Acc No: 2002-446126/200248

Related WPI Acc No: 2003-734495

XRPX Acc No: N02-351506

Compile method involves generating object program by compiling source program, which is stored in object program file along with source program

Patent Assignee: HITACHI LTD (HITA ); HITACHI SOFTWARE ENG CO LTD (HISF )

Inventor: ASAKA S; HOSOTANI H; MITSUMORI M

Number of Countries: 026 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 1202171	A2	20020502	EP 2001106817	A	20010319	200248 B

Priority Applications (No Type Date): JP 2000332110 A 20001025

Patent Details:

Patent No Kind Lan Pg Main IPC Filing Notes

EP 1202171 A2 E 15 G06F-009/45

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT  
LI LT LU LV MC MK NL PT RO SE SI TR

Abstract (Basic): EP 1202171 A2

NOVELTY - An **object program** generated by compiling a **source program**, is stored in an **object program file** (112) along with the source program. The source program is associated with the object program in the object program file.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are included for:

(1) Compiler

(2) Recorded medium storing compile program.

USE - Compile method.

ADVANTAGE - Increases execution speed of **compiled source program** by optimization process. A procedure being compiled can be searched for any change made and **compiled** when necessary. A **source program** can be easily read or it can be encrypted to make it readable. Large proportion of compiling time is saved. Procedures can be updated easily.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of compiler.

Object program file (112)

pp; 15 DwgNo 1/9

Title Terms: COMPILE; METHOD; GENERATE; OBJECT; PROGRAM; COMPILE; SOURCE;  
PROGRAM; STORAGE; OBJECT; PROGRAM; FILE; SOURCE; PROGRAM

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-F07 ; T01-S03

35/9/8 (Item 8 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

014193416 \*\*Image available\*\*

WPI Acc No: 2002-014113/200202

XRPX Acc No: N02-011399

Commands optimization method involves combining commands of source program converted commands into object program to obtain single block of commands

Patent Assignee: FUJITSU LTD (FUIT )

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
JP 2001265605	A	20010928	JP 20013885	A	20010111	200202 B

Priority Applications (No Type Date): JP 20003732 A 20000112

Patent Details:

Patent No Kind Lan Pg Main IPC Filing Notes

JP 2001265605 A 8 G06F-009/45

Abstract (Basic): JP 2001265605 A

NOVELTY - Several blocks of branch instructions of source program (2) transformed into the predetermined commands of object program by compiler, are combined to form single block of commands.

DETAILED DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

(a) Compiler device;

(b) Recording medium storing compiling program;

(c) Compiling program

USE - For transforming a **source program** such as formula translation (FORTRAN) language and common business oriented language (COBOL) into object program.

ADVANTAGE - The number of branch instructions are reduced, as several blocks of commands are combined to form a single block.

DESCRIPTION OF DRAWING(S) - The figure shows the explanatory

diagram of the internal component of a compiler. (Drawing includes non-English language text).

Source program (2)

pp; 8 DwgNo 1/7

Title Terms: COMMAND; METHOD; COMBINATION; COMMAND; SOURCE; PROGRAM;  
CONVERT; COMMAND; OBJECT; PROGRAM; OBTAIN; SINGLE; BLOCK; COMMAND

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F03A; T01-F05A

?t35/9/13-14,17,21-22

35/9/13 (Item 13 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

013741324 \*\*Image available\*\*

WPI Acc No: 2001-225554/200123

XRPX Acc No: N01-160140

**Multiple user program translating apparatus for use in client-server system, has dynamic translator for executing source instruction by translating code block and creating translation file to be accessed by client**

Patent Assignee: HEWLETT-PACKARD CO (HEWP )

Inventor: ISTVAN A F; LE B; PATEL A

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 6158047	A	20001205	US 98111956	A	19980708	200123 B

Priority Applications (No Type Date): US 98111956 A 19980708

Patent Details:

Patent No Kind Lan Pg Main IPC Filing Notes

US 6158047 A 12 G06F-009/45

Abstract (Basic): US 6158047 A

NOVELTY - A source code module (SCM) (16) with several code blocks comprising source instructions are mapped to memory (17) by operating system. Virtual instruction pointer indicates the **source instruction** to be executed. Client/dynamic **translator** (19) executes the instruction by translating a block and storing in buffer (32). A shared translation file is created for translated block, which is accessed by a client.

DETAILED DESCRIPTION - The apparatus translates user program into native code that runs on a native computer hardware that has memory with **source file** including the program to be **translated into native machine code**. The operating system is capable of detecting whether the program has an instruction set architecture different from native instruction set architecture. An INDEPENDENT CLAIM is also included for multiple user program translating method.

USE - For client-server system which performs both static and dynamic compilation.

ADVANTAGE - Enables **translating source program**, as the program executes at run time without terminating the program. Enables **translating all the source code** in the source file as well as the dynamically generated code. Collects, analyzes and periodically submits the profile data to enable optimization which leads to better machine code quality, thus better performance when executing that code is achieved. The apparatus periodically maps new translated code to shared memory so that multiple users simultaneously execute the code.

DESCRIPTION OF DRAWING(S) - The figure shows the block diagram of client-server system.

SCM (16)

Memory (17)

Client/dynamic translator (19)

Buffer (32)

pp; 12 DwgNo 1/4

Title Terms: MULTIPLE; USER; PROGRAM; TRANSLATION; APPARATUS; CLIENT; SERVE

; SYSTEM; DYNAMIC; TRANSLATION; EXECUTE; SOURCE; INSTRUCTION; TRANSLATION  
; CODE; BLOCK; TRANSLATION; FILE; ACCESS; CLIENT

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-J20A; T01-M02A1B

35/9/14 (Item 14 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

013669571 \*\*Image available\*\*

WPI Acc No: 2001-153783/200116

XRPX Acc No: N01-113355

Program management procedure e.g. for source and object program, involves  
judging whether object program for investigation is compiled from  
source program by comparing dump files

Patent Assignee: NEC CORP (NIDE )

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
JP 2000357097	A	20001226	JP 99169426	A	19990616	200116 B

Priority Applications (No Type Date): JP 99169426 A 19990616

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
JP 2000357097	A		6 G06F-009/45	

Abstract (Basic): JP 2000357097 A

NOVELTY - The object program (12) output by compiling source  
program (11) according to compile information (23) extracted from  
object program (21) for investigation is stored along with source  
program in dump files. The dump files are compared to judge whether  
the object program (21) is compiled from source program .

DETAILED DESCRIPTION - The compile information (23) extracted from  
object program for investigation, includes object program name,  
compiler name, compile option, compiler version and compile time.

USE - For managing correlation of source program, object program  
and run program.

ADVANTAGE - Specifies complete correlation between each programs  
easily.

DESCRIPTION OF DRAWING(S) - The figure shows the process flow  
diagram of program management procedure.

Source program (11)

Object programs (12,21)

Compile information (23)

pp; 6 DwgNo 1/3

Title Terms: PROGRAM; MANAGEMENT; PROCEDURE; SOURCE; OBJECT; PROGRAM;

JUDGEMENT; OBJECT; PROGRAM; INVESTIGATE; COMPILE; SOURCE; PROGRAM;

COMPARE; DUMP; FILE

Derwent Class: T01

International Patent Class (Main): G06F-009/45

International Patent Class (Additional): G06F-009/06

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-F06

35/9/17 (Item 17 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

013238556 \*\*Image available\*\*

WPI Acc No: 2000-410430/200035

Related WPI Acc No: 2000-085701; 2000-627603; 2000-627650; 2001-307065;

2001-353573; 2001-388706; 2001-578454

XRPX Acc No: N00-306676

Source program processing method in computer, involves compiling source

program into object program which includes instructions for processing and/or invoking procedures on data field

Patent Assignee: INT BUSINESS MACHINES CORP (IBM )

Inventor: CARTER W A; ELDERON A R; MAGEE T D; NICHOLAS M D; SAADE H Y; SUTHERLAND G; TINDALL W N J; URS J R; WEINMANN T E; WHEATLEY M T

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 6064817	A	20000516	US 97899444	A	19970723	200035 B
			US 97971072	A	19971114	

Priority Applications (No Type Date): US 97971072 A 19971114; US 97899444 A 19970723

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
US 6064817	A	14	G06F-009/45	CIP of application US 97899444

Abstract (Basic): US 6064817 A

NOVELTY - At least one of the programming language statements stored in memory has a data declaration extension requesting a Y2K solution selected from several windowing techniques. The **source program is compiled** into an object program in the memory, where the object program includes instructions for processing and/or invoking procedures on data field associated with declaration according to requested Y2K solution.

DETAILED DESCRIPTION - The data declaration extension syntax is provided. The programming language statements comprising a source program is stored in memory of computer. The data declaration extension provided in programming language statements is in the format of provided data declaration extension syntax. INDEPENDENT CLAIMS are also included for the following:

- (a) computer programming apparatus;
- (b) article of manufacture

USE - For processing source program in computer system to process insurance information, account information, inventory investment and retirement information.

ADVANTAGE - Performs computations directly by reducing required modifications to source code. Allows use of a debugger or other analysis tool at run time, to assist with run time analysis and validation of application, by identifying possible run time conflicts efficiently.

DESCRIPTION OF DRAWING(S) - The figure shows flowchart implementing Y2K solution.

pp; 14 DwgNo 2/2

Title Terms: SOURCE; PROGRAM; PROCESS; METHOD; COMPUTER; COMPILE; SOURCE; PROGRAM; OBJECT; PROGRAM; INSTRUCTION; PROCESS; INVOKE; PROCEDURE; DATA; FIELD

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-F07 ; T01-S03

35/9/21 (Item 21 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

012813621 \*\*Image available\*\*

WPI Acc No: 1999-619852/199953

XRPX Acc No: N99-457141

Propagation of source code locations into objects in compiler of computer system

Patent Assignee: UNISYS CORP (BURS )

Inventor: BAISLEY D E; ZIEBELL J V

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 5978587	A	19991102	US 97969192	A	19971113	199953 B

Priority Applications (No Type Date): US 97969192 A 19971113

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
US 5978587	A		12	G06F-009/45	

Abstract (Basic): US 5978587 A

NOVELTY - For each compiler object, a machine instruction object is constructed and the source object is destroyed. For each machine instruction object, the machine instruction code address is combined with machine instruction object's source object, so as to build a source location table stored in binary file.

DETAILED DESCRIPTION - A source mark object is constructed for each source language element parsed by compiler. Source mapped objects for language element is created and source mark object is destroyed.

USE - For compiling source programs into binary programs for execution on computer system.

ADVANTAGE - Enables encapsulation of source location management in objects that can be readily adapted for use in any compiler using objects.

DESCRIPTION OF DRAWING(S) - The figure shows flow chart explaining propagation of source code locations in computer.

pp; 12 DwgNo 3A, 3B/5

Title Terms: PROPAGATE; SOURCE; CODE; LOCATE; OBJECT; COMPILE; COMPUTER; SYSTEM

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A

35/9/22 (Item 22 from file: 350)

DIALOG(R) File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

012736975 \*\*Image available\*\*

WPI Acc No: 1999-543092/199946

XRFX Acc No: N99-402804

Compiler for translating source program into object program

Patent Assignee: MATSUSHITA ELECTRIC IND CO LTD (MATU ); MATSUSHITA DENKI SANGYO KK (MATU )

Inventor: NAKAJIMA M

Number of Countries: 027 Number of Patents: 004

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 947922	A2	19991006	EP 99106643	A	19990331	199946 B
JP 11345127	A	19991214	JP 9983570	A	19990326	200009
US 6334212	B1	20011225	US 99281812	A	19990331	200206
JP 3264901	B2	20020311	JP 9983570	A	19990326	200220

Priority Applications (No Type Date): JP 9888473 A 19980401

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
EP 947922	A2	E	27	G06F-009/45	

Designated States (Regional): AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI

JP 11345127 A 16 G06F-009/45

US 6334212 B1 G06F-009/45

JP 3264901 B2 18 G06F-009/45 Previous Publ. patent JP 11345127

Abstract (Basic): EP 947922 A2

NOVELTY - The compiler translates a source program into an object program, and minimizes the ultimate code size of an object program that has been translated from a source program including a number of instructions.

DETAILED DESCRIPTION - The compiler calculates a total length of the instructions, in which variables for the source program are



allocated to a first type of register resources in accordance with a first instruction format, and a second instruction length calculator calculates a total length of the instructions, where the variables are allocated to a second type of register resources in accordance with a second instruction format. The length of one instruction defined by the second instruction format is different from that defined by the first instruction format. INDEPENDENT CLAIMS are included for; a system for minimizing the code size of an object program executed on a computer; a computer-readable storage medium storing an object program translated using a compiler from a source program.

USE - Minimizing ultimate code size of object program that has been translated from source program including number of instructions.

ADVANTAGE - Minimizes state of machine-executable object program.

DESCRIPTION OF DRAWING(S) - The drawing shows a block diagram illustrating a configuration of the compiler according to the invention.

pp; 27 DwgNo 1/20

Title Terms: COMPILE; TRANSLATION; SOURCE; PROGRAM; OBJECT; PROGRAM

Derwent Class: T01

International Patent Class (Main): G06F-009/45

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A

?

PLEASE ENTER A COMMAND OR BE LOGGED OFF IN 5 MINUTES

?t35/9/39

35/9/39 (Item 39 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

007303744

WPI Acc No: 1987-300751/198743

XRPX Acc No: N87-224675

Computer system with source code re-creation capability - appends  
compiled code information necessary to re-create source which generated  
compiled code

Patent Assignee: TEXAS INSTR INC (TEXTI )

Inventor: SRIVASTAVA A

Number of Countries: 004 Number of Patents: 002

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
EP 243110	A	19871028	EP 87303392	A	19870416	198743 B
US 5249275	A	19930928	US 86854221	A	19860421	199340
			US 88191857	A	19880504	
			US 89316556	A	19890227	
			US 91696265	A	19910430	

Priority Applications (No Type Date): US 86854221 A 19860421; US 88191857 A 19880504; US 89316556 A 19890227; US 91696265 A 19910430

Cited Patents: 3.Jnl.ref; A3...9122; No-SR.Pub

Patent Details:

Patent No Kind Lan Pg Main IPC Filing Notes

EP 243110 A E 19

Designated States (Regional): DE FR GB

US 5249275 A 8 G06F-009/45 Cont of application US 86854221

Cont of application US 88191857

Cont of application US 89316556

Abstract (Basic): EP 243110 A

The method involves translating a source code statement into an object code block, appending to the block information sufficient to recreate the source code statement, and linking the object code block and appended information into a list with object code and appended information for any related source code statements. The above steps are repeated for each of the sources code statements.

The linking step pref. includes creating a procedure execution frame for each procedure defined by the source code statements, each

frame having pointers, each of which points to a list of object code blocks having a common property, determining which frame corresponds to the procedure in which the source code statement belongs, selecting a list of blocks pointed to by a pointer in the determined frame which have a common property with the source code statements, and inserting the block into the selected list.

ADVANTAGE - In compiling PROLOG programs, allows program statements which use original **source code** to be **compiled**.

Dwg.0/5

Abstract (Equivalent): US 5249275 A

A method enabling the computer to **compile source code** statements of a programming language involves **translating a source code** statement into an object code block and using a compiler of the computer to append to the object code block information sufficient to exactly recreate the source code statement. The object code block and appended information are linked into a list with **object code** and appended information for any related **source code** statements. The **translation, appending** and linking processes are repeated for each of the source code statements.

The linking process involves creating a procedure execution frame for each procedure defined by the source code statements. Each procedure execution frame has a number of pointers, wherein each pointer points to a list of object code blocks having a common property. The linking process also involves determining which procedure execution frame corresponds to the procedure in which the source code statement belongs, selecting a list of object code blocks pointed to by a pointer in the determined procedure execution frame which have a common property with the **source code** statement, and **inserting the object code** block into the selected list.

USE/ADVANTAGE - Provides system usable in programming environment for languages such as PROLOG which allows full compilation of statements such as CLAUSE, ASSERT and RETRACT. Compiles language leaving no portions to be executed by interpreter at execution time.

Dwg.3/6

Title Terms: COMPUTER; SYSTEM; SOURCE; CODE; CREATION; CAPABLE; COMPILE; CODE; INFORMATION; NECESSARY; SOURCE; GENERATE; COMPILE; CODE

Derwent Class: T01

International Patent Class (Main): G06F-009/45

International Patent Class (Additional): G06F-009/44

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05

?t35/9/44

35/9/44 (Item 44 from file: 347)

DIALOG(R)File 347:JAPIO

(c) 2004 JPO & JAPIO. All rts. reserv.

06124314 \*\*Image available\*\*

FILE CONVERTING METHOD AND RECORDING MEDIUM

PUB. NO.: 11-065851 [JP 11065851 A]

PUBLISHED: March 09, 1999 (19990309)

INVENTOR(s): YUSA AKIKAZU

APPLICANT(s): MITSUBISHI ELECTRIC CORP

APPL. NO.: 09-217846 [JP 97217846]

FILED: August 12, 1997 (19970812)

INTL CLASS: G06F-009/45

#### ABSTRACT

PROBLEM TO BE SOLVED: To easily specify what kind of converting process system has generated an object file later and to easily and securely perform file management by adding version information and/or name information on the converting process system used for the **converting** process to a **source file** and the object file.

SOLUTION: When the **source file** is **converted** through the certain **converting** process to generate the object file, a program for actualizing the file converting process for adding information on the day and time of

the **converting** process to the **source file** and **object file** is stored on a disk memory 2 and an arithmetic processor 1 performs the file conversion according to the program. Consequently, even when object files having the same name are generated by changing the converting process system, pieces of conversion day and time information are compared to specify on which source file the object file is generated.

COPYRIGHT: (C)1999,JPO

?

40/9/9 (Item 9 from file: 350)  
DIALOG(R)File 350:Derwent WPIX  
(c) 2004 Thomson Derwent. All rts. reserv.

012727515 \*\*Image available\*\*  
WPI Acc No: 1999-533628/199945  
XRPX Acc No: N99-396333

**Data processing procedure of in-circuit emulator system - involves  
converting in- source code to object code by inserting desired  
object code in arbitrary positions of object program**

Patent Assignee: NEC CORP (NIDE )  
Number of Countries: 001 Number of Patents: 001  
Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
JP 11232091	A	19990827	JP 9829732	A	19980212	199945 B

Priority Applications (No Type Date): JP 9829732 A 19980212

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
JP 11232091	A	21	G06F-009/06	

Abstract (Basic): JP 11232091 A

NOVELTY - The **source code** input by user is **converted** to object code by inserting desired object code in arbitrary positions of object program. A process executing unit (5) reads the object program stored in object holder (7) and verifies the operation at appropriate timing. DETAILED DESCRIPTION - An INDEPENDENT CLAIM is also included for the data processor.

USE - In in-circuit emulator (ICE) system.

ADVANTAGE - Since the object program is verified at appropriate timings, error correction in object program is done quickly. Avoids need for preparing a separate source program. DESCRIPTION OF DRAWING(S) - The figure shows logical structure of ICE system. (5) Process executing unit; (7) Object holder.

Dwg.1/9

Title Terms: DATA; PROCESS; PROCEDURE; CIRCUIT; EMULATION; SYSTEM; CONVERT;  
SOURCE; CODE; OBJECT; CODE; INSERT; OBJECT; CODE; ARBITRARY; POSITION;  
OBJECT; PROGRAM

Derwent Class: T01

International Patent Class (Main): G06F-009/06

International Patent Class (Additional): G06F-011/28

File Segment: EPI

Manual Codes (EPI/S-X): T01-F06; T01-G05A

40/9/10 (Item 10 from file: 350)  
DIALOG(R)File 350:Derwent WPIX  
(c) 2004 Thomson Derwent. All rts. reserv.

012588632 \*\*Image available\*\*  
WPI Acc No: 1999-394739/199933  
XRPX Acc No: N99-295061

**Absolute memory address determining system for intermediate code model**

Patent Assignee: SUN MICROSYSTEMS INC (SUNM )

Inventor: DAMRON P C

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 5920722	A	19990706	US 97933253	A	19970923	199933 B

Priority Applications (No Type Date): US 97933253 A 19970923

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
US 5920722	A	9	G06F-009/445	

Abstract (Basic): US 5920722 A

NOVELTY - A **compiler** (18) **translates** **source code** to **object**

code and instantiate each absolute reference address with a code sequence for referring a subset of address space. Each absolute address in the address space subset forms intermediate code model. An **object code** supply (22) stores the translated source code with instantiated code sequence as **object code** for execution by a processor (13).

**DETAILED DESCRIPTION** - A processor interfacing with memory (12) has several addressable locations, each addressable location is referred by absolute address having a maximum size directly proportional to total number of addressable location in the main memory. A source code supply (20) specifies program routines which has at least one reference to absolute address within the address space. The system (10) further has a code generator (19) within the compiler for instantiating each absolute address reference in the source code program routing and generates relocatable machine code which is stored in relocatable machine code supply (21). The processor has two registers in which the first register stores 22 bits in one half and 10 bits in another half. The second register stores 12 bit in its segment. **INDEPENDENT CLAIMS** are also included for the following:

- (a) process of determining absolute address;
- (b) storage medium for determining absolute memory address

**USE** - To determine absolute memory address for intermediate code model.

**ADVANTAGE** - As the total size of address space is smaller than the maximum size and directly proportional to total number of addressable addresses, absolute address is determined for an intermediate model.

**DESCRIPTION OF DRAWING(S)** - The figure depicts the system for determining absolute memory address.

System (10)  
Memory (12)  
Processor (13)  
Compiler (18)  
Code generator (19)  
Source code supply (20)  
Relocatable machine code (21)  
Object code supply (22)  
pp; 9 DwgNo 1/6

Title Terms: ABSOLUTE; MEMORY; ADDRESS; DETERMINE; SYSTEM; INTERMEDIATE; CODE; MODEL

Derwent Class: T01

International Patent Class (Main): G06F-009/445

File Segment: EPI

Manual Codes (EPI/S-X): T01-F01B; T01-F05B

40/9/11 (Item 11 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

012374447 \*\*Image available\*\*

WPI Acc No: 1999-180554/199915

XRPX Acc No: N99-132604

**Executed method on system for generating presentation object capable of displaying static information dynamically generated on user interface**

Patent Assignee: LUTRIS TECHNOLOGIES INC (LUTR-N)

Inventor: CLARK K; DIEKHANS M E; MORGAN P A

Number of Countries: 082 Number of Patents: 004

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
WO 9908182	A1	19990218	WO 98US16348	A	19980805	199915 B
AU 9886938	A	19990301	AU 9886938	A	19980805	199928
EP 1002267	A1	20000524	EP 98938409	A	19980805	200030
			WO 98US16348	A	19980805	
JP 2001512868	W	20010828	WO 98US16348	A	19980805	200156
			JP 2000506580	A	19980805	

Priority Applications (No Type Date): US 9754817 P 19970805

Patent Details:

Patent No	Kind	Lan	Pg	Main IPC	Filing Notes
WO 9908182	A1	E	31	G06F-007/00	
Designated States (National): AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES FI GB GE GH GM HR HU ID IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG UZ VN YU ZW					
Designated States (Regional): AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW NL OA PT SD SE SZ UG ZW					
AU 9886938	A				Based on patent WO 9908182
EP 1002267	A1	E		G06F-007/00	Based on patent WO 9908182
Designated States (Regional): AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE					
JP 2001512868	W		39	G06F-009/44	Based on patent WO 9908182

Abstract (Basic): WO 9908182 A1

NOVELTY - Object-text source file is received that integrates structured text and object oriented code. The structured text and object-oriented code are converted into integrated object-oriented code module, which is compiled into a presentation object capable of generating structured text from static information and dynamically generated information.

DETAILED DESCRIPTION - User can request web page with static and dynamic information by invoking a presentation object (124) located on a server (101). The object is created by compiling object-oriented code (121) together with structured text (116), such as text formatted using hypertext markup language (HTML) codes that can be used to display static information on the web page. Complex web pages can be developed for use in enterprise networks by combining object-oriented code and structured text.

USE - For generating information on user interface on computer based systems.

ADVANTAGE - Structured text portion can be modified without modifying object-oriented programming language. More sophisticated software developers can use object-oriented programming language to generate information to be used in web page, and complex web pages can be developed for use in enterprise networks.

DESCRIPTION OF DRAWING(S) - Diagram shows exemplary system consistent with system and method of implementing present invention.

Server (101)

Structured text (116)

Object-oriented code (121)

Presentation object. (124)

pp; 31 DwgNo 1/5

Title Terms: EXECUTE; METHOD; SYSTEM; GENERATE; PRESENT; OBJECT; CAPABLE; DISPLAY; STATIC; INFORMATION; DYNAMIC; GENERATE; USER; INTERFACE

Derwent Class: T01; T04

International Patent Class (Main): G06F-007/00; G06F-009/44

International Patent Class (Additional): G06F-009/45 ; G06F-012/00; G06K-007/10

File Segment: EPI

Manual Codes (EPI/S-X): T01-J12; T01-J20A; T04-H03; T04-X

40/9/12 (Item 12 from file: 350)

DIALOG(R) File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

011981590 \*\*Image available\*\*

WPI Acc No: 1998-398500/199834

XRPX Acc No: N98-310057

Source code interprocedural analysis implementing method e.g. for computer software compilation systems - involves invoking separate instance of compiler back end for each IPA output file to produce standard format binary object files which are linked to produce final output

Patent Assignee: SILICON GRAPHICS INC (SILI-N)

Inventor: DEHNERT J C; HIRANANDANI S; HO W W; LEUNG L H

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 5778212	A	19980707	US 96657196	A	19960603	199834 B

Priority Applications (No Type Date): US 96657196 A 19960603

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
US 5778212	A	18	G06F-009/44	

Abstract (Basic): US 5778212 A

The method is applicable to a source code contained in multiple source files. Each source file contains one or more programs. The intermediate representations of the source files are received. Information containing one or more programs of each source file, is summarised. The intermediate representation of the source files and the summarised information are stored in associated extended object format files. Compilation options for each of the source files, is stored in the object format files. Interprocedural analysis functions are performed on each object format file to generate an IPA output file for each object format file.

A separate instance of a compiler back end is invoked for each IPA output file. Using the compilation options, a standard format binary object file is produced for each IPA output file. The standard format binary object file is linked to produce a final output. The IPA functions, invoking and linking operations are performed under the control of a linkage editor.

USE - E.g. for computer software compilation systems.

ADVANTAGE - Ensures faster compilation and linking by parallel processing of intermediate files. Allows flexibility in designing and implementing multiple programs and reuse of software. Provides safe guard against external references.

Dwg.6/8

Title Terms: SOURCE; CODE; ANALYSE; IMPLEMENT; METHOD; COMPUTER; SOFTWARE; COMPILE; SYSTEM; INVOKE; SEPARATE; INSTANCE; COMPILE; BACK; END; OUTPUT; FILE; PRODUCE; STANDARD; FORMAT; BINARY; OBJECT; FILE; LINK; PRODUCE; FINAL; OUTPUT

Derwent Class: T01

International Patent Class (Main): G06F-009/44

File Segment: EPI

Manual Codes (EPI/S-X): T01-F05A ; T01-S01B

?t40/9/15-16

40/9/15 (Item 15 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

011055955 \*\*Image available\*\*

WPI Acc No: 1997-033879/199703

Related WPI Acc No: 1994-010090; 1994-010091; 1994-010092; 1994-010093;

1997-535335; 1998-008325; 1998-250873; 1999-034502; 1999-396974;

2001-488197; 2003-446761; 2003-844877

XRPX Acc No: N97-028639

Method in computer system for binding to source object - involves instantiating moniker object, storing reference to source object in it which is then stored in destination object, invoking binding code

Patent Assignee: MICROSOFT CORP (MICT )

Inventor: ATKINSON R G; JUNG E K; WILLIAMS A S

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
US 5581760	A	19961203	US 92909983	A	19920706	199703 B
			US 9388724	A	19930706	

Priority Applications (No Type Date): US 9388724 A 19930706; US 92909983 A 19920706

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
US 5581760	A	94	G06F-009/44	CIP of application US 92909983

Abstract (Basic): US 5581760 A

The method involves instantiating a moniker object having a moniker class identifier that identifies binding code. The binding code, when invoked, locates and connects to the source object. A reference to the source object is stored in the instantiated moniker object as naming information. A reference to the instantiated moniker object is stored in the destination object. The destination object is accessed, and the reference stored in the destination object is retrieved to the instantiated moniker object.

The binding code is invoked. then, the server code is located and invoked. The server code is requested to connect to the source object. When the server code is invoked, the source object is instantiated. Data is loaded into the instantiated source object. An indication of the instantiated source object with the loaded data is returned to the invoked binding code. The loaded data of the instantiated source object is accessed using the returned indication.

USE/ADVANTAGE - Generates links to source data incorporated in compound document. Interfaces with links to source data in manner which is independent of underlying source data.

Dwg.15/49

Title Terms: METHOD; COMPUTER; SYSTEM; BIND; SOURCE; OBJECT; OBJECT;  
STORAGE; REFERENCE; SOURCE; OBJECT; STORAGE; DESTINATION; OBJECT; INVOKE;  
BIND; CODE

Derwent Class: T01

International Patent Class (Main): G06F-009/44

File Segment: EPI

Manual Codes (EPI/S-X): T01-F07 ; T01-J20A; T01-S01B

40/9/16 (Item 16 from file: 350)

DIALOG(R)File 350:Derwent WPIX

(c) 2004 Thomson Derwent. All rts. reserv.

010604657 \*\*Image available\*\*

WPI Acc No: 1996-101610/199611

XRPX Acc No: N96-085040

**Program management method - by carrying out comparison and contrast of source program counter saved in object file, and loading module file**

Patent Assignee: MITSUBISHI ELECTRIC CORP (MITQ )

Number of Countries: 001 Number of Patents: 001

Patent Family:

Patent No	Kind	Date	Applicat No	Kind	Date	Week
JP 8006767	A	19960112	JP 94140454	A	19940622	199611 B

Priority Applications (No Type Date): JP 94140454 A 19940622

Patent Details:

Patent No	Kind	Lan Pg	Main IPC	Filing Notes
JP 8006767	A	6	G06F-009/06	

Abstract (Basic): JP 8006767 A

The method involves storing a source program counter in an object file (3). A source program counter is stored in a load module file (5).

A comparison and contrast of the source program counter stored in object file and load module file is carried out.

ADVANTAGE - Achieves source level debugging function.

Dwg.1/3

Title Terms: PROGRAM; MANAGEMENT; METHOD; CARRY; COMPARE; CONTRAST; SOURCE;  
PROGRAM; COUNTER; SAVE; OBJECT; FILE; LOAD; MODULE; FILE

Derwent Class: T01

International Patent Class (Main): G06F-009/06

International Patent Class (Additional): G06F-009/45 ; G06F-011/28

File Segment: EPI

Manual Codes (EPI/S-X): T01-J20C

?t40/9/45

40/9/45 (Item 45 from file: 347)



DIALOG(R)File 347:JAPIO  
(c) 2004 JPO & JAPIO. All rts. reserv.

02175132    \*\*Image available\*\*  
TEST METHOD FOR PROGRAM

PUB. NO.:        62-092032 [JP 62092032 A]  
PUBLISHED:      April 27, 1987 (19870427)  
INVENTOR(s):    KODAMA KIYOSHI  
                 FUKUSHIMA SHIRO  
APPLICANT(s):   OMRON TATEISI ELECTRONICS CO [000294] (A Japanese Company or  
                 Corporation), JP (Japan)  
APPL. NO.:      60-232523 [JP 85232523]  
FILED:          October 17, 1985 (19851017)  
INTL CLASS:     [4] G06F-011/28; G06F-009/44  
JAPIO CLASS:    45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)  
JOURNAL:        Section: P, Section No. 621, Vol. 11, No. 299, Pg. 56,  
                 September 29, 1987 (19870929)

#### ABSTRACT

PURPOSE: To test the execution of a single program by executing this program then an instruction including an address allocated automatically to display them, and making the instruction accessible to the corresponding program.

CONSTITUTION: A debugging system comprises a work station 11, a terminal equipment 12 and a debug executing device 13. When an operation test is executed for the debug, etc. after a **source program** is **compiled**, an address set previously is automatically allocated to an instruction containing an unfixed address to be substituted even in case such instruction is included in a **machine word program** owing to a fact that the **source program** includes an undefined part. Thus the **machine word program** can be executed singly. If the instruction having its executed, this executed instruction is displayed on a display device 27 and a direct access is possible to the program via an emulator 29. Thus an execution test can be carried out with a single program.  
?t40/9/47-49

40/9/47        (Item 47 from file: 347)

DIALOG(R)File 347:JAPIO  
(c) 2004 JPO & JAPIO. All rts. reserv.

02035046    \*\*Image available\*\*  
INTERRUPTION CAUSE CHECKING METHOD

PUB. NO.:        61-249146 [JP 61249146 A]  
PUBLISHED:      November 06, 1986 (19861106)  
INVENTOR(s):    MIYATA KAZUO  
APPLICANT(s):   FUJITSU LTD [000522] (A Japanese Company or Corporation), JP  
                 (Japan)  
APPL. NO.:      60-092124 [JP 8592124]  
FILED:          April 26, 1985 (19850426)  
INTL CLASS:     [4] G06F-011/00; G06F-009/44; G06F-009/46  
JAPIO CLASS:    45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)  
JOURNAL:        Section: P, Section No. 561, Vol. 11, No. 100, Pg. 30, March  
                 28, 1987 (19870328)

#### ABSTRACT

PURPOSE: To point out automatically positions of interruption causes without having the influence upon the processing efficiency by providing a correspondence map file, where correspondence addresses between a source program and an object program are stored, and an error processing part in a restoration control part.

CONSTITUTION: A restoration control part 1 of a CPU consists of a correspondence map file 2, an error processing part 3, an interruption accepting part 4, and a dump processing part 5, and interruption accepted by the accepting part 4 are analyzed by the processing part 3 to perform

processings. The processing part 5 saves a trouble record where an error occurs at the error occurrence time, and the **source program** is stored in the file 2 in accordance with the **machine language** of the object program when the **source program** is interpreted to obtain the object program. When an interruption occurs in the CPU, the accepting part 4 operates the processing part 3, and the file 2 is retrieved on the basis of the address, where the trouble occurs, in case of a program interruption, and a corresponding program name or the like is taken out and is displayed on a CRT 6. Thus, positions of interruption causes are pointed out automatically without having the influence upon the processing efficiency.

40/9/48 (Item 48 from file: 347)  
DIALOG(R)File 347:JAPIO  
(c) 2004 JPO & JAPIO. All rts. reserv.

01991941 \*\*Image available\*\*  
EXTENSION METHOD FOR PROGRAM LANGUAGE

PUB. NO.: 61-206041 [JP 61206041 A]  
PUBLISHED: September 12, 1986 (19860912)  
INVENTOR(s): MARUYAMA SADANOBU  
SUZUKI NAOYA  
APPLICANT(s): SONY CORP [000218] (A Japanese Company or Corporation), JP  
(Japan)  
APPL. NO.: 60-047813 [JP 8547813]  
FILED: March 11, 1985 (19850311)  
INTL CLASS: [4] G06F-009/44  
JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)  
JOURNAL: Section: P, Section No. 543, Vol. 11, No. 37, Pg. 17,  
February 04, 1987 (19870204)

#### ABSTRACT

PURPOSE: To obtain an extension method for a program language which can add easily a new instruction and to use it immediately by assembling the file of a new instruction to a program language itself when the program language starts to operate.

CONSTITUTION: From a keyboard 2 to a **compiler** 1, a **source program** is given, **compiled** successively to an **object program** (ObjP), and the **source program** is successively stored to a memory 4A and the compiled ObjP is successively stored to a memory area 4B respectively. For the program execution including the new instruction, the program of a disk 5 is loaded and a name file 8 is retrieved. Thus, the file name of modules 9 and 10 to add the new additional instruction is known. For that reason, the corresponding module 9 is accessed, and assembled to the program language itself. In the same manner, the module 10 is accessed and assembled to the program language itself. Hereinafter, the retrieval of the name file 8 is continued, the file to add all the additional instructions is assembled to the program language itself and thereafter, execution is performed.

40/9/49 (Item 49 from file: 347)  
DIALOG(R)File 347:JAPIO  
(c) 2004 JPO & JAPIO. All rts. reserv.

01991938 \*\*Image available\*\*  
PROGRAMMING EXECUTING METHOD FOR COMPUTER

PUB. NO.: 61-206038 [JP 61206038 A]  
PUBLISHED: September 12, 1986 (19860912)  
INVENTOR(s): SUZUKI NAOYA  
MARUYAMA SADANOBU  
APPLICANT(s): SONY CORP [000218] (A Japanese Company or Corporation), JP  
(Japan)  
APPL. NO.: 60-047809 [JP 8547809]  
FILED: March 11, 1985 (19850311)  
INTL CLASS: [4] G06F-009/44

JAPIO CLASS: 45.1 (INFORMATION PROCESSING -- Arithmetic Sequence Units)  
JOURNAL: Section: P, Section No. 543, Vol. 11, No. 37, Pg. 16,  
February 04, 1987 (19870204)

ABSTRACT

PURPOSE: To use a common variable with plural programs by storing a variable and information concerning the variable onto the memory and delivering the variable on the disk.

CONSTITUTION: A source program of a BASIC language is given to a compiler 1 from a keyboard 2. The program is compiled successively into an **object program** by a **compiler 1**, the **source program** is **stored** to an area 4A on a memory 4 and the **object program** is edited and stored into an area 4B respectively. On the memory 4, a memory area 4C to form a symbol table of the variable and the information concerning the variable is provided. The variable and the information concerning the variable stored at the area 4C can be easily saved on the disk. Consequently, the variable and the information concerning the variable can be easily delivered from one program to other program.

?

File 348:EUROPEAN PATENTS 1978-2004/Mar W01

(c) 2004 European Patent Office

File 349:PCT FULLTEXT 1979-2002/UB=20040311,UT=20040304

(c) 2004 WIPO/Univentio

Set	Items	Description
S1	59	SOURCECOD?
S2	482484	SOURCE OR HUMANREAD? OR HUMAN(1W)READ????
S3	9682	S2(1W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCEDURE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? OR F- ILE OR FILES OR SUBFILE? ?)
S4	2386	S2(1W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? OR - LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S5	181	S2(2N)COMMENT?
S6	4588	S2(2N)OBJECT? ?
S7	807413	OBJECT OR MACHINEREAD? OR MACHINE OR MACHINELANGUAGE? OR N- ATIVE OR NONHUMAN OR NON(1W)HUMAN
S8	1428	POSITION? ?(1W)(RELOCAT? OR INDEPENDENT)
S9	14018	S7:S8(2W)(CODE? ? OR CODING? ? OR MICROCOD???? ? OR PROCED- URE? ? OR SUBPROCEDURE? OR INSTRUCTION? ? OR SUBINSTRUCTION? ? OR FILE OR FILES OR SUBFILE? ?)
S10	11981	S7:S8(2W)(PROGRAMME? ? OR PROGRAMMING? ? OR PROGRAM???? ? - OR LANGUAGE? ? OR ROUTINE? OR SUBROUTINE? OR MICROFILE?)
S11	7	S7:S8(2W)SUBPROGRAM?
S12	2391	BYTECOD???? ? OR BYTE(1W)(CODE? ? OR CODING? ?)
S13	3	S2(1W)SUBPROGRAM?
S14	979	(S1 OR S3:S6 OR S13)(3N)(STORE? ? OR STORED OR STORING OR - STORAGE OR EMBED? OR IMBED? OR INSET? OR INLAY? OR INLAID? OR INFIX? OR INSERT?)
S15	2013	(S1 OR S3:S6 OR S13)(3N)(INCLUD? OR INCLUSION? OR TOGETHER OR INCORPORAT? OR ATTACH? OR INTEGRAT? OR APPEND? OR COMBIN??? ?)
S16	1428	(S1 OR S3:S6 OR S13)(3N)INCLUD??? ?
S17	24	OBJECTCOD? OR OBJECTFIL?
S18	350	S14:S16(25N)(S9:S12 OR S17)
S19	13	S18/TI,AB
S20	913	IC='G06F-009/45'
S21	237	S14:S16(10N)(S9:S12 OR S17)
S22	45	S21/TI,AB,CM
S23	29	S20 AND S21
S24	64	S19 OR S22:S23
S25	54	S24 NOT OBJECT(1W)ORIENT?? ?(1W)PROGRAM????? ?
S26	54	IDPAT (sorted in duplicate/non-duplicate order)
S27	54	IDPAT (primary/non-duplicate records only)

27/5,K/1 (Item 1 from file: 348)

DIALOG(R)File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

01619532

**Apparatus, method and program for breakpoint setting**

**Einrichtung, Verfahren und Programm für Haltepunkteinstellung**

**Appareil, procede et programme pour definir des points d'arret**

**PATENT ASSIGNEE:**

MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD., (216883), 1006, Oaza-Kadoma,  
Kadoma-shi, Osaka 571-8501, (JP), (Applicant designated States: all)

**INVENTOR:**

Kawai, Masaki, Hiramachipakuhaitu 301, 3-38-1, Hira-machi, Seto-shi,  
Aichi-Ken 489-0916, (JP)

Kawamoto, Takuji, 2-16, Syumoku-cho, Higashi-ku, Nagoya-shi, Aichi-ken  
461-0014, (JP)

**LEGAL REPRESENTATIVE:**

Crawford, Andrew Birkby et al (29762), A.A. Thornton & Co. 235 High  
Holborn, London WC1V 7LE, (GB)

PATENT (CC, No, Kind, Date): EP 1335292 A2 030813 (Basic)

APPLICATION (CC, No, Date): EP 2003250765 030206;

PRIORITY (CC, No, Date): JP 200231372 020207

DESIGNATED STATES: AT; BE; BG; CH; CY; CZ; DE; DK; EE; ES; FI; FR; GB; GR;

HU; IE; IT; LI; LU; MC; NL; PT; SE; SI; SK; TR  
EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO  
INTERNATIONAL PATENT CLASS: G06F-011/36

ABSTRACT EP 1335292 A2

Disclosed is a breakpoint setting apparatus capable of setting a breakpoint without imposing any burden on a programmer. The breakpoint setting apparatus includes an edited-line list manager 115 for managing an address of an edited line in a source code, and a breakpoint setting/disabling subunit 106 for setting a breakpoint at the address stored in the edited-line list manager 115. The breakpoint setting apparatus automatically sets a breakpoint on each line where the programmer makes an edit without any specific instruction from the programmer.

ABSTRACT WORD COUNT: 85

NOTE:

Figure number on first page: NONE

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 030813 A2 Published application without search report  
LANGUAGE (Publication,Procedural,Application): English; English; English  
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	200333	1212
SPEC A	(English)	200333	8595
Total word count - document A			9807
Total word count - document B			0
Total word count - documents A + B			9807

...CLAIMS A2

1. A breakpoint setting apparatus comprising:
  - a loading unit operable to load an **object code** generated correspondingly to a **source code** ;
  - a **storing** unit operable to store information showing relation between each of components constituting the source code...

...each control block.

14. A breakpoint setting method comprising:
  - a loading step of loading an **object code** generated correspondingly to a **source code** ;
  - a **storing** step of storing information showing relation between each of components constituting the source code and...

...the editing information.

15. A breakpoint setting program comprising:
  - a loading step of loading an **object code** generated correspondingly to a **source code** ;
  - a **storing** step of storing information showing relation between each of components constituting the source code and...

27/5,K/2 (Item 2 from file: 348)

DIALOG(R)File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

01423124

Compile method and program recording medium

Kompilierverfahren und - Programmaufzeichnungsmedium

Methode de compilation et media d'enregistrement de programme

PATENT ASSIGNEE:

Hitachi, Ltd., (204145), 6 Kanda Surugadai 4-chome, Chiyoda-ku, Tokyo  
101-8010, (JP), (Applicant designated States: all)

HITACHI SOFTWARE ENGINEERING CO., LTD., (678781), 81, Onoecho-6-chome  
Naka-ku, Yokohama, (JP), (Applicant designated States: all)

INVENTOR:

Mitsumori, Masato, c/o Hitachi, Ltd., 5-1, Marunouchi 1-chome, Chiyoda-ku  
, Tokyo 100-8220, (JP)

Asaka, Shinji, c/o Hitachi, Ltd., 5-1, Marunouchi 1-chome, Chiyoda-ku,

Tokyo 100-8220, (JP)

Hosotani, Hiroyuki, c/o Software Eng., Co., Ltd., 6-81, Onoe-cho, Naka-ku  
, Yokohama-shi, Kanagawa 231--0015, (JP)

LEGAL REPRESENTATIVE:

Strehl Schubel-Hopf & Partner (100941), Maximilianstrasse 54, 80538  
Munchen, (DE)

PATENT (CC, No, Kind, Date): EP 1202171 A2 020502 (Basic)

APPLICATION (CC, No, Date): EP 2001106817 010319;

PRIORITY (CC, No, Date): JP 2000332110 001025

DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI;  
LU; MC; NL; PT; SE; TR

EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI

INTERNATIONAL PATENT CLASS: G06F-009/45

ABSTRACT EP 1202171 A2

A compiler (102), that generates an object program file (112) from a source program (101) in which a plurality of procedures are written, compiles procedures (func1, func2), by regarding the procedures as source-program compile units (602, 803, 804), to generate corresponding **object -program** compile units (601, 801, 802). A plurality of **object -program** compile units generated are output to a memory (106) **together** with the corresponding **source - program** compile units (602, 803, 804). When compiling a source program (101) in which one procedure has been changed, the compiler (102) compiles only the source-program compile unit corresponding to the changed procedure.

ABSTRACT WORD COUNT: 98

NOTE:

Figure number on first page: 1

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 020502 A2 Published application without search report

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	200218	1722
SPEC A	(English)	200218	3576
Total word count - document A			5298
Total word count - document B			0
Total word count - documents A + B			5298

INTERNATIONAL PATENT CLASS: G06F-009/45

...ABSTRACT by regarding the procedures as source-program compile units (602, 803, 804), to generate corresponding **object -program** compile units (601, 801, 802). A plurality of **object - program** compile units generated are output to a memory (106) **together** with the corresponding **source - program** compile units (602, 803, 804). When compiling a source program (101) in which one procedure...

...SPECIFICATION unit corresponding to that source-program compile unit, and means for storing a plurality of **object - program** compile units **together** with the **source program** in one **object - program file**. Note that even when a plurality of source program files with respective names are compiled collectively, procedures changed may be identified by comparing a source program file with a corresponding **source program file** stored in the **object program file** with regard to each of the source program files without checking the file date.  
Further...

...this time. Further, the compiler updates the source information regarding the procedure stored in the **object - program file** to new source information. This new source information includes the updated **source - program** compile unit and information used to analyze the syntax of the source program input to...will be described later, is carried out on all procedures. If there exists a corresponding **object program file** 112, a decision is made whether or not a **source program**, which was stored at its previous compiling, exists in the **object program file** 112 (step 202). If the result of decision at the step 202 is that

the...file 101 are compiled, object-program compile units generated by compiling are stored in the **object program file 112**, and corresponding **source - program** compile units are stored associated with the **object - program** compile units in the **object program file 112**.

In the above example, a decision is made whether or not a source program file 101 specified as undergoing a compile process has its antecedent **source program** stored in the **object program file 112** even when it is compiled for the first time. However, in such a case ...

...be adapted to decide whether or not an instruction is given by a user to **store a source program** in the **object program file** (step 501). If there is given an instruction to **store a source program**, as described above, a source-program compile unit corresponding to the object-program compile unit is output to the **object program file 112**. This makes it possible to prevent an unnecessary **source program** from being **stored** in the **object program file**. A source program of text format may be compressed so as not to be read compiled, and the **object - program** compile unit compiled and a corresponding **source - program** compile unit are **stored** procedure by procedure in the **object program file**. When a source program is updated, normally, constants or variables referred to are often updated...

...object program part 1121 includes an object-program compile unit 801 for func1 and an **object - program** compile unit 802, and the **source program** part 1122 includes a **source - program** compile unit 803 for func1 and a source-program compile unit 804 for func2, stored...

...CLAIMS the steps of:

storing an object program, generated by compiling a source program, in an **object program file**; and  
storing said **source program** corresponding to said **object program** in said **object program file**, said source program being associated with said object program in said object program file.

2...

...in said source program since the previous compiling;  
compiling the changed portion of said input **source program**; and  
storing said **object program** generated by compiling and the changed portion of said input source program in said object...

...information since previous compiling;  
compiling the changed portion of said source information of the input **source program**; and  
storing said **object program** and the changed portion of said source information in said object program file (112).

5...

...or 3, further comprising the steps of:  
inputting an instruction specifying whether or not to **store** said **source program** or said source information in said **object program file** (112); and  
if an instruction not to store is input, storing said object program in ...

27/5,K/3 (Item 3 from file: 348)  
DIALOG(R) File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

01269949

Method and apparatus for compiling program for parallel processing, and computer readable recording medium recorded with parallelizing compilation program

Verfahren und Gerat zum Übersetzen eines Programms fur parallele Verarbeitung, und mit einem parallelisierenden Übersetzungsprogramm beschriebenes rechnerlesbares Aufzeichnungsmedium

Methode et appareil pour compiler un programme pour traitement parallele,

et medium d'enregistrement lisible par ordinateur contenant un  
programme de compilation parallélisante

PATENT ASSIGNEE:

NEC CORPORATION, (236690), 7-1, Shiba 5-chome, Minato-ku, Tokyo, (JP),  
(Applicant designated States: all)

INVENTOR:

Obata, Masaya, NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo,  
(JP)

LEGAL REPRESENTATIVE:

VOSSIUS & PARTNER (100314), Siebertstrasse 4, 81675 Munchen, (DE)

PATENT (CC, No, Kind, Date): ~~EP 1094387 A2 010425~~ (Basic)

APPLICATION (CC, No, Date): EP 122314 001020;

PRIORITY (CC, No, Date): JP 99301316 991022

DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI;  
LU; MC; NL; PT; SE

EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI

INTERNATIONAL PATENT CLASS: G06F-009/45

ABSTRACT EP 1094387 A2

A parallelizing compilation apparatus for generating object codes that can execute processing, which begin at the branch target where control transfers with higher probability, in advance of the execution of a conditional branch instruction in parallel with the processing prior to the conditional branch instruction without the rearrangement of basic blocks is provided. A branch dualizing section (13) determines, based on profile information (17), the truth probability of the evaluation value of the conditional expression in a conditional branch instruction included in intermediate codes. When the probability of "false" is higher, the branch dualizing section dualizes the conditional branch instruction into a conditional branch instruction whose conditional expression is the inversion of that in the dualized conditional branch instruction and whose branch target is the next instruction of the dualized conditional branch instruction. Conversely, when the probability of "true" is higher, the branch dualizing section inserts an unconditional branch instruction just after the dualized conditional branch instruction and sets the branch target thereof to the next instruction of this unconditional branch instruction. A branch inverting section (19) generates object codes in which the target address of conditional branch instructions and unconditional branch instructions are exchanged, when the determination relating to the truth probability using profile information is inverted with respect to that at the time of the generation of an object code file (16).

ABSTRACT WORD COUNT: 225

NOTE:

Figure number on first page: 1

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 010425 A2 Published application without search report

Assignee: 030502 A2 Transfer of rights to new applicant: NEC  
Electronics Corporation (4260580) 1753  
Shimonumabe, Nakahara-ku Kawasaki, Kanagawa  
211-8668 JP

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	200117	1593
SPEC A	(English)	200117	7966
Total word count - document A			9559
Total word count - document B			0
Total word count - documents A + B			9559

INTERNATIONAL PATENT CLASS: G06F-009/45

...SPECIFICATION the third information.

In FIG. 11, an assembly processing section 91 reads out an assembler  
source program stored in an object code file 16, and then  
assembles the program to generate a load module 92. In the assembly...



27/5,K/6 (Item 6 from file: 348)  
DIALOG(R)File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

01076685

Compiler

Kompiler

Compilateur

PATENT ASSIGNEE:

Matsushita Electric Industrial Co., Ltd., (1855508), 1006, Oaza-Kadoma,  
Kadoma-shi, Osaka 571-8501, (JP), (Applicant designated States: all)

INVENTOR:

Nakajima, Masaitsu, 2-10-24-604, Seiiku, Joto-ku, Osaka-shi, Osaka  
536-0007, (JP)

LEGAL REPRESENTATIVE:

Grunecker, Kinkeldey, Stockmair & Schwanhausser Anwaltssozietat (100721)  
, Maximilianstrasse 58, 80538 Munchen, (DE)

PATENT (CC, No, Kind, Date): EP 947922 A2 991006 (Basic)

EP 947922 A3 030625

APPLICATION (CC, No, Date): EP 99106643 990331;

PRIORITY (CC, No, Date): JP 9888473 980401

DESIGNATED STATES: AT; BE; CH; CY; DE; DK; ES; FI; FR; GB; GR; IE; IT; LI;  
LU; MC; NL; PT; SE

EXTENDED DESIGNATED STATES: AL; LT; LV; MK; RO; SI

INTERNATIONAL PATENT CLASS: G06F-009/45

ABSTRACT EP 947922 A2

A compiler is adapted to minimize the ultimate code size of an **object program** that has been translated from a **source program** including a plurality of instructions. The compiler includes first instruction length calculator for calculating a total length of the instructions where variables for the source program are allocated to a first type of register resources in accordance with a first instruction format and second instruction length calculator for calculating a total length of the instructions where the variables are allocated to a second type of register resources in accordance with a second instruction format. The length of one instruction defined by the second instruction format is different from that defined by the first instruction format. The variables are allocated to respectively appropriate ones of the register resources based on the results of calculation derived by the first and second instruction length calculators.

ABSTRACT WORD COUNT: 145

NOTE:

Figure number on first page: 1

LEGAL STATUS (Type, Pub Date, Kind, Text):

Search Report: 030625 A3 Separate publication of the search report

Application: 991006 A2 Published application without search report

Examination: 031001 A2 Date of request for examination: 20030801

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
----------------	----------	--------	------------

CLAIMS A	(English)	9940	646
----------	-----------	------	-----

SPEC A	(English)	9940	6143
--------	-----------	------	------

Total word count - document A	6789
-------------------------------	------

Total word count - document B	0
-------------------------------	---

Total word count - documents A + B	6789
------------------------------------	------

INTERNATIONAL PATENT CLASS: G06F-009/45

...ABSTRACT A2

A compiler is adapted to minimize the ultimate code size of an **object program** that has been translated from a **source program** including a plurality of instructions. The compiler includes first instruction length calculator for calculating a total...

...SPECIFICATION instruction format.

Specifically, a compiler according to the present invention is adapted to translate a **source program**, including a plurality of instructions, into an **object program**. The compiler includes: first instruction length calculating means for calculating a total length of the...means.

A computer-readable storage medium according to the present invention has stored thereon an **object program** that has been translated using a compiler from a **source program** including a plurality of instructions. The **object program** includes not only instructions described in a first instruction format using a first type of...

CLAIMS 1. A compiler for translating a **source program**, including a plurality of instructions, into an **object program**, the compiler comprising:

first instruction length calculating means for calculating a total length of the...

...second instruction length calculating means.

10. A computer-readable storage medium having stored thereon an **object program** that has been translated using a compiler from a **source program** including a plurality of instructions,

wherein the **object program** includes not only instructions described in a first instruction format using a first type of...

27/5,K/9 (Item 9 from file: 348)

DIALOG(R) File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

00988185

Logic analyzer for identifying an acquisition sample corresponding to a source code statement

Logikanalysator zur Erkennung der Ubereinstimmung einer Abtastungsprobe mit einem Quellcodebefehl

Analyseur logique pour l'identification d'un echantillon correspondant a une commande de code source

PATENT ASSIGNEE:

TEKTRONIX, INC., (1893633), 26600 S.W. Parkway, P.O. Box 1000,  
Wilsonville, Oregon 97070, (US), (Proprietor designated states: all)

INVENTOR:

Heath, Robert J., 2225 SW 187th, Aloha, Oregon 97006, (US)

LEGAL REPRESENTATIVE:

Burke, Steven David et al (47741), R.G.C. Jenkins & Co. 26 Caxton Street,  
London SW1H 0RJ, (GB)

PATENT (CC, No, Kind, Date): EP 893762 A1 990127 (Basic)  
EP 893762 B1 030521

APPLICATION (CC, No, Date): EP 98305714 980717;

PRIORITY (CC, No, Date): US 895544 970717

DESIGNATED STATES: DE; FR; GB

INTERNATIONAL PATENT CLASS: G06F-011/00

CITED PATENTS (EP B): EP 261247 A; EP 317080 A

CITED REFERENCES (EP B):

"CodeView And Utilities" 1987, MICROSOFT CORPORATION, REDMOND, WA  
XP002084741 \* Chapter 7, page 158, line 1-5 \*;

ABSTRACT EP 893762 A1

A system and method for locating a data sample in a logic analyzer or emulator acquisition buffer corresponding to a source code statement (58) employs first (56) and second (54) windows, wherein one window (56) displays the contents of the acquisition buffer, for example, in disassembly form and the other window (54) displays source code. When a source code file is displayed and a specified statement is selected within the source code, a search (36) for a matching sample in the acquisition buffer is initiated. If a sample is found, it is displayed and highlighted (40) in the appropriate window.

ABSTRACT WORD COUNT: 101

NOTE:

Figure number on first page: 6

LEGAL STATUS (Type, Pub Date, Kind, Text):

Examination: 020109 A1 Date of dispatch of the first examination  
report: 20011122  
Application: 990127 A1 Published application (A1with Search Report  
;A2without Search Report)  
Grant: 030521 B1 Granted patent  
Change: 021120 A1 Title of invention (French) changed: 20021002  
Change: 021120 A1 Title of invention (English) changed: 20021002  
Change: 021120 A1 Title of invention (German) changed: 20021002  
Change: 021218 A1 Title of invention (German) changed: 20021025  
Change: 021218 A1 Title of invention (English) changed: 20021025  
Change: 021218 A1 Title of invention (French) changed: 20021025  
Examination: 990616 A1 Date of filing of request for examination:  
990420

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	199904	683
CLAIMS B	(English)	200321	478
CLAIMS B	(German)	200321	434
CLAIMS B	(French)	200321	568
SPEC A	(English)	199904	2843
SPEC B	(English)	200321	2855
Total word count - document A			3527
Total word count - document B			4335
Total word count - documents A + B			7862

...CLAIMS said means (56) for viewing a display of instruction lines of a  
program compiled into **object code** from said program written in  
said **source code** including a second cursor positioning means  
for highlighting a line of **object code** ; and

further characterized in that when said search means locates an  
occurrence of said acquired...

?t27/5,k/15,17,20

27/5,K/15 (Item 15 from file: 348)

DIALOG(R)File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

00826139

Method and apparatus for analyzing software executed in embedded systems  
Verfahren und Anordnung zur Analyse von in eingebetteten Systemen  
ausfuehrender Software

Methode et appareil pour l'analyse de logiciel executant dans des systemes  
encastres

PATENT ASSIGNEE:

Applied Microsystems, Inc., (2192570), 5020 148th Avenue N.E., Redmond,  
WA 98052, (US), (applicant designated states: DE;FR;GB;SE)

INVENTOR:

Rees, Andrew John, 1402 NE 65th Street, Seattle, Washington 98115, (US)  
O'Brien, Stephen Caine, 3907 - 204th Avenue NE, Redmond, Washington 98053  
, (US)  
Krystad, Peter D., 17526 Fremont Avenue North, Seattle, Washington 98133,  
(US)

LEGAL REPRESENTATIVE:

Grunecker, Kinkeldey, Stockmair & Schwanhausser Anwaltssozietat (100721)  
, Maximilianstrasse 58, 80538 Munchen, (DE)

PATENT (CC, No, Kind, Date): EP 767430 A2 970409 (Basic)  
EP 767430 A3 990120

APPLICATION (CC, No, Date): EP 96114557 960911;

PRIORITY (CC, No, Date): US 526709 950911

DESIGNATED STATES: DE; FR; GB; SE

INTERNATIONAL PATENT CLASS: G06F-011/00; G06F-011/34;

ABSTRACT EP 767430 A2

A software analysis system for capturing tags generated by tag statements in instrumented source code. The software analysis system includes a probe that monitors the address and data bus of the target system. When a tag statement is executed in the target system, a tag is written to a predetermined location in the address space of the target system. The tag contains a tag value that is indicative of the location in the source code of the tag statement generating the tag. By monitoring the predetermined address, the probe is able to capture tags as they are written on the data bus of the target system. By properly instrumenting the source code, the software analysis system is able to perform a variety of analysis functions in essentially real time, including code coverage, function and task execution times, memory allocation, call pairs, and program tracing.

ABSTRACT WORD COUNT: 145

LEGAL STATUS (Type, Pub Date, Kind, Text):

Refusal: 020424 A2 Date European patent application was refused:  
20011123  
Application: 970409 A2 Published application (A1with Search Report  
;A2without Search Report)  
Search Report: 990120 A3 Separate publication of the European or  
International search report  
Change: 990127 A2 Obligatory supplementary classification  
(change)  
Examination: 990728 A2 Date of filing of request for examination:  
990527  
Examination: 991229 A2 Date of dispatch of the first examination  
report: 19991110

LANGUAGE (Publication,Procedural,Application): English; English; English  
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB97	2080
SPEC A	(English)	EPAB97	8700
Total word count - document A			10780
Total word count - document B			0
Total word count - documents A + B			10780

...CLAIMS code of said software, and wherein said method further includes the steps of compiling said source code and inserted tag statements to obtain object code, and executing said object code in said target system.

27/5,K/17 (Item 17 from file: 348)

DIALOG(R) File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

00797673

A program translating apparatus and a processor which achieve high-speed execution of subroutine branch instructions

Programmubersetzungsgerat und Prozessor, die eine schnelle Ausfuhrung von Unterprogrammsprungbefehlen erreichen

Appareil de traduction de programmes et processeur achevant l'execution a haute vitesse d'instructions de branchement a des sous-programmes

PATENT ASSIGNEE:

MATSUSHITA ELECTRIC INDUSTRIAL CO., LTD., (216882), 1006, Kadoma,  
Kadoma-shi, Osaka-fu 571, (JP), (Proprietor designated states: all)

INVENTOR:

Takayama, Shuichi, 1-22-6, Nakayamadai, Takarazuka-shi, Hyogo 665, (JP)

Higaki, Nobuo, Shaining-Sato 2H, 4-15-26, Komatsu, Higashiyodogawa-ku,  
Osaka-shi, Osaka 533, (JP)

Tominaga, Nobuki, Form-Fushimimomoyama 501, 215-1, Higashi-machi,  
Fushimi-ku, Kyoto-shi, Kyoto 612, (JP)

Mijayi, Shinya, 3-11-1, Jinguu, Nara-shi, Nara 631, (JP)

Urushibara, Seiichi, 5-15 Fukakusakawakubo-cho, Fukushima-ku, Kyoto-shi,  
Kyoto, (JP)

LEGAL REPRESENTATIVE:

Crawford, Andrew Birkby et al (29761), A.A. Thornton & Co. 235 High

Holborn, London WC1V 7LE, (GB)  
 PATENT (CC, No, Kind, Date): EP 742517 A2 961113 (Basic)  
 EP 742517 A3 970528  
 EP 742517 B1 010801  
 APPLICATION (CC, No, Date): EP 96301214 960223;  
 PRIORITY (CC, No, Date): JP 95111701 950510  
 DESIGNATED STATES: DE; FR; GB; NL  
 INTERNATIONAL PATENT CLASS: G06F-009/45 ; G06F-009/445; G06F-009/32  
 CITED PATENTS (EP B): US 5375242 A  
 CITED REFERENCES (EP B):  
 MICROPROCESSORS AND MICROSYSTEMS, JULY-AUG. 1990, UK, vol. 14, no. 6,  
 ISSN 0141-9331, pages 385-396, XP000151094 MCGEADY S: "Inside Intel's  
 i960CA superscalar processor"  
 MICROPROCESSING AND MICROPROGRAMMING, vol. 36, no. 5, 1 October 1993,  
 pages 243-257, XP000397907 GONZALEZ A M: "A SURVEY OF BRANCH TECHNIQUES  
 IN PIPELINED PROCESSORS";

ABSTRACT EP 742517 A2

A program translating apparatus is composed of a translation unit 103 and a link unit 108. The translation unit 103 includes a determination unit 105 which detects the stack size to be needed for each subroutine included in a source program to be translated into a machine instruction sequence and the name of a register to be retrieved in the process of each subroutine. The determination unit 105 then stores the stack size and the name detected into a file together with the machine instruction sequence. The link unit 108 includes the following units: A branch instruction detection unit 109 detects a branch instruction from the machine instruction sequence when machine instruction sequences stored in different files are linked each other. A file detection unit 110 and an acquisition unit 111 retrieve the stack size and the register name from the file which has the branch target subroutine. A subroutine call instruction generation unit 112 replaces the branch instruction with an instruction which consequently executes a branch operation, a stack reservation, and register retrieval. (see image in original document)

ABSTRACT WORD COUNT: 205

NOTE:

Figure number on first page: 5

LEGAL STATUS (Type, Pub Date, Kind, Text):

Examination: 001213 A2 Date of dispatch of the first examination report: 20001031  
 Application: 961113 A2 Published application (A1with Search Report ;A2without Search Report)  
 Oppn None: 020724 B1 No opposition filed: 20020503  
 Grant: 010801 B1 Granted patent  
 Lapse: 020502 B1 Date of lapse of European Patent in a contracting state (Country, date): FR 20011228,  
 Change: 970521 A2 Obligatory supplementary classification (change)  
 Search Report: 970528 A3 Separate publication of the European or International search report  
 Examination: 971203 A2 Date of filing of request for examination: 971009

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS A	(English)	EPAB96	1138
CLAIMS B	(English)	200131	1141
CLAIMS B	(German)	200131	1042
CLAIMS B	(French)	200131	1503
SPEC A	(English)	EPAB96	4891
SPEC B	(English)	200131	4891
Total word count - document A			6031
Total word count - document B			8577
Total word count - documents A + B			14608

...ABSTRACT a determination unit 105 which detects the stack size to be needed for each subroutine included in a source program to be translated into a machine instruction sequence and the name of a register to be retrieved in the process of each...

...SPECIFICATION into a machine instruction sequence, the stack size necessary for the process of each subroutine included in the source program is detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are... sequence, the name of a register to be saved in the process of each subroutine included in the source program is detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are...

...and the name of a register to be saved in the process of each subroutine included in the source program are detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are...

...sequences. The translation unit 103 receives and processes the input files 101 and 102 where source programs are stored, and outputs the output files 106 and 107 where machine instruction sequences are stored.

The translation unit translates a file as a unit. For example, if...

...SPECIFICATION into a machine instruction sequence, the stack size necessary for the process of each subroutine included in the source program is detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are... sequence, the name of a register to be saved in the process of each subroutine included in the source program is detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are...

...and the name of a register to be saved in the process of each subroutine included in the source program are detected and stored in a file together with the machine instruction sequence of the corresponding subroutine.

Then, when the machine instruction sequences in different files are...

...sequences. The translation unit 103 receives and processes the input files 101 and 102 where source programs are stored, and outputs the output files 106 and 107 where machine instruction sequences are stored.

The translation unit translates a file as a unit. For example, if...

27/5,K/20 (Item 20 from file: 348)

DIALOG(R) File 348:EUROPEAN PATENTS

(c) 2004 European Patent Office. All rts. reserv.

00597960

Programmable computer with automatic translation between source and object code with version control

Programmierbarer Rechner mit automatischer Übersetzung zwischen Quell - und Zielcode mit Versionsüberwachung

Ordinateur programmable avec traduction automatique entre code source et code-cible avec controle de version

PATENT ASSIGNEE:

AMDAHL CORPORATION, (628802), 1250 East Arques Avenue, Sunnyvale, CA 94088, (US), (applicant designated states:

AT;BE;CH;DE;DK;ES;FR;GB;IT;LI;LU;NL;SE)

INVENTOR:

Knudsen, Helge, 46 Eighth Concession Road, RR#1, Freelton, Ontario L0R 1K0, (CA)

Chong, Daniel T., 2 Frankie Lane, Woodbridge, Ontario L4L 7J6, (CA)

Yaffe, John, 1239 Tenth Concession Road West, RR#3, Puslinch, Ontario N0B

2J0, (CA)

Taughner, James, 1504 Chalice Crescent, Mississauga, Ontario L5C 1S3, (CA)

Robertson, Michael, 2641 Burnford Trail, Mississauga, Ontario L5M 4E1, (CA)

Plazak, Zbigniew, 70 Glen Agar, Etobicoke, Ontario M9B 5M1, (CA)

LEGAL REPRESENTATIVE:

Crawford, Andrew Birkby et al (29761), A.A. THORNTON & CO. Northumberland

House 303-306 High Holborn, London WC1V 7LE, (GB)

PATENT (CC, No, Kind, Date): EP 588446 A2 940323 (Basic)

EP 588446 A3 951115

EP 588446 B1 990707

APPLICATION (CC, No, Date): EP 93203242 900904;

PRIORITY (CC, No, Date): US 402862 890901; US 450298 891213

DESIGNATED STATES: AT; BE; CH; DE; DK; ES; FR; GB; IT; LI; LU; NL; SE

RELATED PARENT NUMBER(S) - PN (AN):

EP 489861 (EP 909143406)

INTERNATIONAL PATENT CLASS: G06F-009/44; G06F-009/45 ; G06F-017/30

ABSTRACT EP 588446 A2

A computer which executes rules which are defined according to a language having a valid grammar. The computer comprises input means for receiving and temporarily storing a first source code representation of a rule; object code translation means for translating the first source code representation into a first object code representation executable by the computer; storage means for storing the object code representations of rules; discard means for automatically discarding from the input means the first source code representation upon the storing of the first object code representation in the storage means; source code translation means for translating the first object code representation into a second source code representation where the second source code representation has lines of text; edit means for editing the second source code representation by deleting, adding, or changing one or more of the lines of text of the second source code representation; second object code translation means for translating the second source code representation, as edited, into a new object code representation of the edited rule for storage in the storage means; and the discard means discarding the first object code and second source code representations automatically upon the storing of the new object code representation of the edited rule in the storage means. The computer thereby minimizes the storage required in the storage means for storing rules and maintains version control over the object code representations of rules stored in the storage means.

A method for manipulating a database of data and rules stored in a computer system where the computer operates in accordance with object-coded rules defined by a specified object code grammar. The computer including storage means for storing data and object-coded rules in tables in conformance with a storage architecture, control means for storing, retrieving and deleting data and object-coded rules from the tables, translator means for translating source-coded rules into object-coded rules, detranslator means for translating object-coded rules into source-coded rules and scanner means for determining lexical validity of an object - coded rule according to the grammar. The method first comprising the ordered steps of entering into the computer a first source - coded rule; storing the first source - coded rule into tables in the storage means; translating the first source-coded rule into a first object - coded rule; storing the first object - coded rule into the tables in the storage means; and discarding the first source-coded rule from the tables in the storage means. Secondly, the method comprises the steps of retrieving an object - coded rule from the tables in the storage means; translating the object - coded rule into a second source-coded rule; editing the second source - coded rule; storing the second source - coded rule, as edited, into the tables in the storage means; translating the second source-coded rule

into a second **object - coded** rule; storing the second **object - coded** rule into the tables in the storage means and discarding the original object-coded rule and the second source-coded rule from the tables in the storage means. The method further requires that each translating step determines lexical validity or invalidity of the **object - coded** rule translated from the source-coded rule; that each storing step conditions the storage of an object-coded rule in the tables in the storage means upon the determination of validity of the object-coded rule to be stored and each discarding step conditions the discarding of the first source-coded rule upon the storing of said first object-coded rule into the tables in the storage means and of the original **object - coded** rule and the second **source - coded** rule upon the **storing** of the second **object - coded** rule in the tables in the storage means.

ABSTRACT WORD COUNT: 594

LEGAL STATUS (Type, Pub Date, Kind, Text):

Lapse: 000614 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707,

Application: 940323 A2 Published application (Alwith Search Report ;A2without Search Report)

Lapse: 031105 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707, CH 19990707, LI 19990707, DK 19991007, NL 19990707, SE 19990707,

Lapse: 020605 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707, CH 19990707, LI 19990707, SE 19990707,

Lapse: 001213 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707, CH 19991012, LI 19991012,

Oppn None: 000628 B1 No opposition filed: 20000408

Lapse: 001227 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707, CH 19990707, LI 19990707,

Lapse: 030219 B1 Date of lapse of European Patent in a contracting state (Country, date): AT 19990707, BE 19990707, CH 19990707, LI 19990707, NL 19990707, SE 19990707,

Examination: 940323 A2 Date of filing of request for examination: 931208

Change: 940615 A2 Inventor (change)

Search Report: 951115 A3 Separate publication of the European or International search report

Examination: 970514 A2 Date of despatch of first examination report: 970326

Change: 980722 A2 International patent classification (change)

Change: 980722 A2 Obligatory supplementary classification (change)

Change: 990707 A2 Title of invention (French) (change)

Grant: 990707 B1 Granted patent

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9927	1636
CLAIMS B	(German)	9927	1403
CLAIMS B	(French)	9927	1808
SPEC B	(English)	9927	16336
Total word count - document A			0
Total word count - document B			21183
Total word count - documents A + B			21183

...INTERNATIONAL PATENT CLASS: G06F-009/45



...ABSTRACT a language having a valid grammar. The computer comprises input means for receiving and temporarily storing a first source code representation of a rule; object code translation means for translating the first source code representation into a first object code representation executable by the computer; storage means for storing the object code representations of rules; discard means for automatically discarding from the input means the first source code representation upon the storing of the first object code representation in the storage means; source code translation means for translating the first object code representation into a second source code representation where the second source code representation has lines...

...coded rules into source-coded rules and scanner means for determining lexical validity of an object - coded rule according to the grammar. The method first comprising the ordered steps of entering into the computer a first source - coded rule; storing the first source - coded rule into tables in the storage means; translating the first source-coded rule into a first object - coded rule; storing the first object - coded rule into the tables in the storage means; and discarding the first source-coded rule...

...the tables in the storage means. Secondly, the method comprises the steps of retrieving an object - coded rule from the tables in the storage means; translating the object - coded rule into a second source-coded rule; editing the second source - coded rule; storing the second source - coded rule, as edited, into the tables in the storage means; translating the second source-coded rule into a second object - coded rule; storing the second object - coded rule into the tables in the storage means and discarding the original object-coded rule...

...The method further requires that each translating step determines lexical validity or invalidity of the object -coded rule translated from the source-coded rule; that each storing step conditions the storage...

...first object-coded rule into the tables in the storage means and of the original object - coded rule and the second source - coded rule upon the storing of the second object - coded rule in the tables in the storage means.

...SPECIFICATION only one single version or representation of any given program on the computer. Only the object code version of a program is stored on the secondary storage medium. The source code version is only generated on demand by the detranslation from the object code. The object...

?t27/5,k/23,25-26

27/5,K/23 (Item 23 from file: 348)  
DIALOG(R)File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

00487120

Arrangement for efficiently transferring program execution between subprograms

Vorrichtung zur wirksamen Übertragung von Programmabläufen zwischen Unterprogrammen

Dispositif de transfert efficace d'execution de programme entre sous-programmes

PATENT ASSIGNEE:

AT&T Corp., (589370), 32 Avenue of the Americas, New York, NY 10013-2412, (US), (applicant designated states: DE;ES;FR;GB;IT;SE)

INVENTOR:

DeBruler, Dennis L., 4720 Main Street, Downers Grove, Illinois 60515, (US)

LEGAL REPRESENTATIVE:

Watts, Christopher Malcolm Kelway, Dr. et al (37392), Lucent Technologies (UK) Ltd, 5 Mornington Road, Woodford Green Essex IG8 OTU, (GB)

PATENT (CC, No, Kind, Date): EP 474425 A2 920311 (Basic)  
EP 474425 A3 920617  
EP 474425 B1 980603

APPLICATION (CC, No, Date): EP 91307861 910828;

PRIORITY (CC, No, Date): US 577427 900904

DESIGNATED STATES: DE; ES; FR; GB; IT; SE

INTERNATIONAL PATENT CLASS: G06F-009/40;

CITED REFERENCES (EP A):

PROCEEDINGS OF THE 1987 I.E.E.E. INTERNATIONAL CONFERENCE ON COMPUTER  
DESIGN: VLSI IN COMPUTERS

I.E.E.E. MICRO vol. 8, no. 3, June 1988, NEW YORK, NY, US pages 63 - 76;

D. P. RYAN: 'Intel's 80960: An Architecture Optimised for Embedded  
Control'

MICROPROCESSORS AND MICROSYSTEMS vol. 14, no. 6, August 1990, LONDON, GB  
pages 385 - 396; S. MCGEADY: 'Inside Intel's i960CA superscalar  
processor';

ABSTRACT EP 474425 A2

An arrangement called PASS CONTROL (FIG. 11) is used in combination  
with a conventional RETURN statement as a substitute for a conventional  
CALL-and-RETURN subprogram invocation sequence (FIG. 2), and effects a  
return from a whole series of subprogram invocations directly to the  
subprogram that initiated the series without intervening returns to the  
subprograms that made the intermediate invocations in the series. The  
arrangement uses the conventional execution stack (114) to effect the  
series of invocations and the return therefrom (FIGS. 12-14). The  
subprograms that are invoked by the series of invocations share an  
execution stack frame (1620). Both a compiler arrangement and an  
application program execution arrangement for effecting PASS CONTROL  
functionality are disclosed. (see image in original document)

ABSTRACT WORD COUNT: 121

LEGAL STATUS (Type, Pub Date, Kind, Text):

Lapse: 040121 B1 Date of lapse of European Patent in a  
contracting state (Country, date): ES  
19981217, IT 19980603, SE 19980903,  
Application: 920311 A2 Published application (A1with Search Report  
;A2without Search Report)  
Search Report: 920617 A3 Separate publication of the European or  
International search report  
Examination: 930203 A2 Date of filing of request for examination:  
921207  
\*Assignee: 940622 A2 Applicant (name, address) (change)  
Examination: 970502 A2 Date of despatch of first examination report:  
970313  
Grant: 980603 B1 Granted patent  
Lapse: 990113 B1 Date of lapse of the European patent in a  
Contracting State: SE 980903  
Oppn None: 990526 B1 No opposition filed  
Lapse: 991020 B1 Date of lapse of European Patent in a  
contracting state (Country, date): IT  
19980603, SE 19980903,

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9823	2064
CLAIMS B	(German)	9823	2089
CLAIMS B	(French)	9823	2178
SPEC B	(English)	9823	7378
Total word count - document A			0
Total word count - document B			13709
Total word count - documents A + B			13709

...CLAIMS source program (110) comprising a plurality of different source  
subprograms (200-209) made up of source code statements  
including subprogram invocation statements and subprogram return  
statements into an object program (112) comprising a plurality of  
object subprograms made up of object instructions, the means

including...

- ...invoking a source subprogram, for generating an instruction to branch (1216) from execution of an **object subprogram** compiled from the **source subprogram** that includes the invoking statement, to execution of an **object subprogram** compiled from the invoked source subprogram,  
second means responsive to encountering (1200) in a source...
  - ...a pointer (703) to an instruction at which is to resume the execution of an **object subprogram** compiled from the **source subprogram** (200) that includes the first invocation statement, and  
third means responsive to encountering a return statement (501) in...
  - ...instruction pointer (703) from the execution stack (114) and to branch from execution of an **object subprogram** (209) compiled from the **source subprogram** that includes the returnstatement directly to execution of the **object subprogram** (200) compiled from the **source subprogram** that includes the first invocation statement in the series at the instruction pointed to by the instruction...
  - ...703) retrieved from the execution stack (114), without in a meantime resuming execution of any **object subprogram** compiled from a **source subprogram** (201-208) that includes an intermediate invocation statement in the series, CHARACTERISED BY  
the means for compiling refraining from...
  - ...instruction to store (307) on the execution stack (114) an execution context (704) of the **object subprogram** compiled from the **source subprogram** (200) that includes the first invocation statement; wherein  
the means for compiling refrain from generating instructions to store...
  - ...503) execution context (704) from the execution stack (114) to the execution context of the **object subprogram** compiled from the **source subprogram** (200) that includes the first invocation statement (CALL A), without generating instructions to restore in a meantime the execution context to an execution context of any **object subprogram** compiled from the **source subprogram** (201-208) that includes an intermediate invocation statement (PASS CTRL) in the series.
12. The arrangement of claim 9...

27/5,K/25 (Item 25 from file: 348)  
DIALOG(R)File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

00353786

Method and apparatus for monitoring the execution time of a computer-executed object programme

Verfahren zur Beobachtung des zeitlichen Ablaufs eines von einem Rechnersystem ausgeführten Objektprogrammes und Beobachtungswerkzeug zur Durchführung dieses Ve

Methode et appareil d'observation du deroulement dans le temps d'un programme objet realise par un systeme d'ordinateur

PATENT ASSIGNEE:

ASEA BROWN BOVERI AG, (956641), Haselstrasse 16, 5400 Baden, (CH),  
(applicant designated states: CH;DE;FR;GB;IT;LI;NL;SE)

INVENTOR:

Danuser, Andreas, Oberriedenstrasse 32A, CH-5412 Gebenstorf, (CH)  
Krings, Lothar, Dr., Landliweg 6, CH-5400 Baden, (CH)

PATENT (CC, No, Kind, Date): EP 368190 A1 900516 (Basic)  
EP 368190 B1 970917

APPLICATION (CC, No, Date): EP 89120470 891106;

PRIORITY (CC, No, Date): CH 884161 881109

DESIGNATED STATES: CH; DE; FR; GB; IT; LI; NL; SE

INTERNATIONAL PATENT CLASS: G06F-011/00;

CITED REFERENCES (EP A):

- IBM TECHNICAL DISCLOSURE BULLETIN, Band 30, Nr. 6, November 1987, Seiten 296-297, Armonk, New York, US; "Performance trace facility"
- IDEM
- IBM TECHNICAL DISCLOSURE BULLETIN, Band 26, Nr. 11, April 1984, Seiten 6217-6220, Armonk, New York, US; C.P. GEER et al.: "Instruction stream trace"
- ELECTRONIC DESIGN, Nr. 22, 19. September 1985, Seiten 117-131, Hayden Publishing Co., Inc., Hasbrouck Heights, New Jersey, US; B. ABLEIDINGER et al.: "Real-time analyzer furnishes high-level look at software operation"
- IFORMATIK SPEKTRUM, Band 8, Nr. 1, Seiten 37-38, Springer-Verlag, Berlin, DE; R. KLAR: "Hardware/Software-Monitoring"
- IDEM;

ABSTRACT EP 368190 A1 (Translated)

The execution time of a computer-executed object program (15) is monitored with the method. In this, information relating to the execution of the object program (15) is recorded and evaluated at an interface (16) of the computer system by a monitoring tool (4). This method is intended to permit monitoring of even complex computer systems virtually under real-time conditions with comparatively simple means. This is achieved by the following measures:

Unambiguously identifiable monitoring points in the form of output commands are inserted into the source program (3) associated with the object program (15) at unambiguously localised points.

The monitoring points are stored in a database (11, 12) of the monitoring tool in table form specifying the program points.

During the execution of the program, identifiers associated with the monitoring points are sent by the computer system to the monitoring tool.

The monitoring tool (4) forms events, specifying the current time and the identification of the sending object computer (1) of the computer system.

The events formed are evaluated in the monitoring tool (4) by reference to the monitoring points stored in table form in the language of the source program (3).

TRANSLATED ABSTRACT WORD COUNT: 194

ABSTRACT EP 368190 A1

Mit dem Verfahren wird der zeitliche Ablauf eines von einem Rechnersystem ausgeführten Objektprogramms (15) beobachtet. Hierbei werden an einer Schnittstelle (16) des Rechnersystems von einem Beobachtungswerkzeug (4) den Ablauf des Objektprogramms (15) betreffende Informationen erfasst und ausgewertet - Dieses Verfahren soll mit vergleichsweise einfachen Mitteln die Beobachtung selbst komplexer Rechnersysteme nahezu unter Echtzeitbedingungen ermöglichen. Dies wird durch folgende Massnahmen erreicht:

In das dem Objektprogramm (15) zugeordnete Quellprogramm (3) werden an eindeutig lokalisierten Stellen eindeutig identifizierbare Beobachtungspunkte in Form von Ausgabebefehlen eingefügt.

Die Beobachtungspunkte werden unter Angabe der Programmstellen tabellarisch in einer Datenbank (11, 12) des Beobachtungswerkzeugs gespeichert.

Bei Ablauf des Programms werden vom Rechnersystem den Beobachtungspunkten zugeordnete Kennungen an das Beobachtungswerkzeug gesendet.

Das Beobachtungswerkzeug (4) bildet unter Angabe der aktuellen Zeit und der Identifizierung des sendenden Objektrechners (1) des Rechnersystems Ereignisse.

Die gebildeten Ereignisse werden durch Bezug auf die tabellarisch gespeicherten Beobachtungspunkte in der Sprache des Quellprogramms (3) im Beobachtungswerkzeug (4) ausgewertet.

ABSTRACT WORD COUNT: 156

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 900516 A1 Published application (A1with Search Report  
;A2without Search Report)

Examination: 901227 A1 Date of filing of request for examination:  
901026  
Examination: 940330 A1 Date of despatch of first examination report:  
940217  
\*Assignee: 940810 A1 Applicant (transfer of rights) (change): ASEA  
BROWN BOVERI AG (956641) Haselstrasse 16  
CH-5401 Baden (CH) (applicant designated  
states: CH;DE;FR;GB;IT;LI;NL;SE)

Grant: 970917 B1 Granted patent  
Oppn None: 980909 B1 No opposition filed  
LANGUAGE (Publication,Procedural,Application): German; German; German  
FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9709W2	801
CLAIMS B	(German)	9709W2	632
CLAIMS B	(French)	9709W2	760
SPEC B	(German)	9709W2	2093

Total word count - document A 0

Total word count - document B 4286

Total word count - documents A + B 4286

...ABSTRACT by the following measures:

Unambiguously identifiable monitoring points in the form of output  
commands are inserted into the source program (3) associated with  
the object program (15) at unambiguously localised points.

The monitoring points are stored in a database (11, 12...

27/5,K/26 (Item 26 from file: 348)  
DIALOG(R) File 348:EUROPEAN PATENTS  
(c) 2004 European Patent Office. All rts. reserv.

00337734

Programmable controller with stored tokenized source code  
Speicherprogrammierbare Steuerung mit gespeichertem markierten Quellencode  
Automate programmable avec code source marque et memorise  
PATENT ASSIGNEE:

ALLEN-BRADLEY COMPANY, INC., (204331), 1201 South Second Street,  
Milwaukee Wisconsin 53204, (US), (Proprietor designated states: all)

INVENTOR:

Flood, Mark A., 1330 Worton Avenue, Mayfield Heights Ohio 44124, (US)  
Rischar, Charles M., 98 Larchwood Drive, Painsville Ohio 44077, (US)  
Toma, Jack F., 7404 West Pleasant Valley Road, Parma Ohio 44130, (US)  
Kalan, Michael D., 1245 Jackie Lane, Mayfield Heights Ohio 44124, (US)  
Sepsi, Robert R., 145 Chestnut Lane No. 324G, Richmond Heights Ohio 44143  
, (US)

LEGAL REPRESENTATIVE:

Lippert, Hans, Dipl.-Ing. et al (7783), Holtz Martin Lippert  
Emil-Claar-Strasse 20, 60322 Frankfurt am Main, (DE)

PATENT (CC, No, Kind, Date): EP 331060 A2 890906 (Basic)  
EP 331060 A3 900912  
EP 331060 B1 950823  
EP 331060 B2 990811

APPLICATION (CC, No, Date): EP 89103362 890225;

PRIORITY (CC, No, Date): US 161484 880229

DESIGNATED STATES: DE; FR; GB; IT

INTERNATIONAL PATENT CLASS: G05B-019/05

CITED PATENTS (EP A): DE 3610433 A; US 4449180 A; EP 97444 A; US 4488258 A;  
FR 2473766 A

CITED PATENTS (EP B): EP 97444 A; EP 236828 A; DE 3610433 A; FR 2473766 A;  
US 4449180 A; US 4488258 A

ABSTRACT EP 331060 A2

A programmable controller executes a compiled version of a ladder  
diagram type control program to control the functions of a piece of  
equipment. The compiled program includes not only the machine language  
instructions but also a tokenized version of the source code which was  
used to generate various portions of the machine language instructions  
that cannot be easily used to regenerate the source code and ladder

diagram. This facilitates the editing of the program as the original ladder diagram may be recreated from a combination of the object code and a tokenized version of the source code.

ABSTRACT WORD COUNT: 101

LEGAL STATUS (Type, Pub Date, Kind, Text):

Application: 890906 A2 Published application (A1with Search Report  
;A2without Search Report)  
Search Report: 900912 A3 Separate publication of the European or  
International search report  
Examination: 910227 A2 Date of filing of request for examination:  
901222  
Examination: 930602 A2 Date of despatch of first examination report:  
930416  
Grant: 950823 B1 Granted patent  
\*Assignee: 950906 B1 Proprietor of the patent (transfer of rights):  
ALLEN-BRADLEY COMPANY, INC. (204331) 1201 South  
Second Street Milwaukee Wisconsin 53204 (US)  
(applicant designated states: DE;FR;GB;IT)  
\*Assignee: 950906 B1 Previous applicant in case of transfer of  
rights (change): Allen-Bradley Company (204330)  
1201 South Second Street Milwaukee Wisconsin  
53204 (US) (applicant designated states:  
DE;FR;GB;IT)  
Oppn: 960724 B1 Opposition 01/960522 Siemens AG; Postfach 22 16  
34; D-80506 Munchen; (DE)  
Change: 980930 B1 Representative (change)  
Change: 990506 B1 International patent classification (change)  
Amended: 990811 B2 Amended patent  
Amended: 990811 B2 Date of patent maintained as amended: 19990811

LANGUAGE (Publication,Procedural,Application): English; English; English

FULLTEXT AVAILABILITY:

Available Text	Language	Update	Word Count
CLAIMS B	(English)	9932	650
CLAIMS B	(German)	9932	542
CLAIMS B	(French)	9932	736
SPEC B	(English)	9932	9157
Total word count - document A			0
Total word count - document B			11085
Total word count - documents A + B			11085

?t27/5,k/30-31,36,40

27/5,K/30 (Item 30 from file: 349)

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

01081424 \*\*Image available\*\*

PROCESS FOR COMPILING AND EXECUTING SOFTWARE APPLICATIONS IN A  
MULTI-PROCESSOR ENVIRONMENT  
PROCEDE DE COMPILEATION ET D'EXECUTION D'APPLICATIONS LOGICIELLES DANS UN  
ENVIRONNEMENT MULTIPROCESSEUR

Patent Applicant/Assignee:

SOSPITA AS, OKSA 18, N-4505 Mandal, NO, NO (Residence), NO (Nationality),  
(For all designated states except: US)

Patent Applicant/Inventor:

CARLSEN ULF, Kvanneid, N-4770 Hovag, NO, NO (Residence), NO (Nationality)  
, (Designated only for: US)

HAMMERSTAD Hakon, Victorias vei 16, N-4515 Mandal, NO, NO (Residence), NO  
(Nationality), (Designated only for: US)

GORANCIC Emir, Oksevollen, N-4514 Mandal, NO, NO (Residence), NO  
(Nationality), (Designated only for: US)

Legal Representative:

BRYN AARFLOT AS (agent), P.O.Box 449, N-0104 Oslo, NO,

Patent and Priority Information (Country, Number, Date):

Patent: WO 200403861 A1 20040108 (WO 0403861)

Application: WO 2003NO167 20030522 (PCT/WO NO2003000167)

Priority Application: NO 20023194 20020701

Designated States: AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU

CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP  
KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NI NO NZ OM PH PL PT  
RO RU SC SD SE SG SK SL TJ TM TN TR TT TZ UA UG US UZ VC VN YU ZA ZM ZW  
(EP) AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LU MC NL PT RO SE  
SI SK TR  
(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG  
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZM ZW  
(EA) AM AZ BY KG KZ MD RU TJ TM

Main International Patent Class: G07F-007/10

International Patent Class: G06F-009/44; G06K-019/073

Publication Language: English

Filing Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 7584

#### English Abstract

The present invention relates to multi-application, secure operating systems for small, secure devices, such as smart card microcontrollers. In particular, the present invention relates to mechanisms for secure runtime upload of applications onto small devices, authorisation mechanisms and the ability for authorised execution of multiple applications on the devices, where an application may be potentially larger than the microcontroller memory size. The mechanism simplifies life-cycle smart card management aspects related to post-issuance application ("applet") upload and upgrade. Mechanisms to prepare applications (i.e. compiler techniques) using a common set of project files in one compiler toolset, for execution in a dual host & chip processor environment are described. These help automising the programming of the communication interfaces between the host and chip applications. An important motivation for the present invention is to provide a secure co-processor environment for general computer applications in order to counter software piracy, and to allow new models for secure electronic software distribution and software licensing.

#### French Abstract

La presente invention concerne des systemes d'exploitation securises multi-applications pour petits dispositifs securises, tels que des microcontrolleurs a carte intelligente. En particulier, la presente invention concerne des mecanismes de telechargement en amont d'executions securisees d'applications sur des petits dispositifs, des mecanismes d'autorisation et la capacite d'execution autorisee d'applications multiples sur les dispositifs, une application pouvant etre potentiellement plus grande que la taille de la memoire du microcontrolleur. Le mecanisme simplifie les aspects de gestion de cycle de vie de carte intelligente associes au telechargement en amont et a la mise a jour de l'application (<= applet >=) apres son emission. Sont egalement decrits des mecanismes pour preparer des applications (par exemple des techniques de compilation) utilisant un ensemble commun de dossiers de projet dans un ensemble d'outils de compilation, pour l'execution dans un environnement a deux processeurs hote et puce. Ces mecanismes aident a l'automatisation de la programmation des interfaces de communication entre les applications hote et puce. Un objectif important de la presente invention est de creer un environnement a coprocesseurs securise pour des applications informatiques generales pour contrer le piratage de logiciels, et egalement pour permettre l'utilisation de nouveaux modeles pour la distribution de logiciels et l'octroi de licences pour logiciels electroniques securises.

Legal Status (Type, Date, Text)

Publication 20040108 A1 With international search report.

Fulltext Availability:

Claims

#### Claim

... method of claim 21 further comprising:

(c) re-compiling the precompiled version of the original source code into a single integrated executable machine code program having function calls which are associated with the encrypted executable machine code.

24 A method...

bad date

27/5,K/31 (Item 31 from file: 349)  
DIALOG(R)File 349:PCT FULLTEXT  
(c) 2004 WIPO/Univentio. All rts. reserv.

01047087 \*\*Image available\*\*

**DISPLAYABLE PRESENTATION PAGE**

**PAGE DE PRESENTATION AFFICHABLE**

Patent Applicant/Assignee:

THOUGHT INC, 657 Mission Street, San Francisco, CA 94105, US, US  
(Residence), US (Nationality), (For all designated states except: US)

Patent Applicant/Inventor:

MULLINS Ward, 2222 Leavenworth Street, Apartment 304, San Francisco, CA  
94133, US, US (Residence), US (Nationality), (Designated only for: US)

Legal Representative:

ORZECZOWSKI Karen Lee (agent), Liniak, Berenato & White, LLC, 6550 Rock  
Spring Drive, Suite 240, Bethesda, MD 20817, US,

Patent and Priority Information (Country, Number, Date):

Patent: WO 200377123 A1 20030918 (WO 0377123)

Application: WO 2003US6346 20030303 (PCT/WO US0306346)

Priority Application: US 2002361795 20020304

Designated States: AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU

CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP

KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ OM PH PL PT RO

RU SC SD SE SG SK SL TJ TM TN TR TT TZ UA UG US UZ VC VN YU ZA ZM ZW

(EP) AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LU MC NL PT RO SE

SI SK TR

(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG

(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZM ZW

(EA) AM AZ BY KG KZ MD RU TJ TM

Main International Patent Class: G06F-009/44

International Patent Class: G06G-005/00; G06F-007/00

Publication Language: English

Filing Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 18673

**English Abstract**

Systems, methods and software for creating or maintaining distributed transparent persistence of complex data objects and associated data store and an application programming object capable of creating or maintaining distributed transparent persistence of data objects or data object graphs without the necessity of inserting any byte codes or modification of the object graph. Virtually any java object or object praph can be transparently persisted. Further, copies of a data graph or of a portion of the data graph can be automatically reconciled and changes persisted without any persistence coding in the object model.

**French Abstract**

L'invention concerne des systemes, des procedes et un logiciel destines a creer ou assurer une persistance transparente repartie d'objets de donnees complexes et de memoires de donnees associees. Dans un aspect, l'invention concerne egalement un objet de programmation d'application capable de creer ou d'assurer une persistance transparente repartie d'objets de donnees ou de graphes d'objets de donnees sans qu'il soit necessaire d'insérer des pseudo-codes binaires ou une modification du graphe d'objet. Pratiquement n'importe quel objet Java ou graphe d'objet peut etre sauvegarde de maniere transparente. Par ailleurs, les copies d'un graphe de donnees ou d'une partie du graphe de donnees peuvent etre



automatiquement rapprochees et les changements sauvegardes sans codage persistant dans le modele objet.

Legal Status (Type, Date, Text)

Publication 20030918 A1 With international search report.

Publication 20030918 A1 Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

Fulltext Availability:

Claims

Claim

... a CDOG model, comprising:

a) a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, wherein the set of definitions is stored in a repository;

b) a set of...an object model, by utilizing a displayable presentation page format wherein the displayable presentation page source code contains embedded object programming code, and persisting one or more of a member selected from the group consisting of an...a software module for creating a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, wherein the software module is capable of causing the storage of

the set of...based upon access to (a) a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, (b) a set of definitions for the relationships between objects for an object language ...to relational mapping resource, said software modules comprising:

a). at least one presentation page having embedded within its source code object language programming logic that references attributes of a data object and also referencing at least one associated...

27/5,K/36 (Item 36 from file: 349)

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00920191 \*\*Image available\*\*

METHODS AND APPARATUS FOR ENABLING LOCAL JAVA OBJECT ALLOCATION AND COLLECTION

PROCEDES ET APPAREIL POUR PERMETTRE L'ALLOCATION ET LA COLLECTE D'OBJETS JAVA LOCAUX

Patent Applicant/Assignee:

SUN MICROSYSTEMS INC, M/S: UPAL01-521, 901 San Antonio Road, Palo Alto, CA 94303, US, US (Residence), US (Nationality)

Inventor(s):

WALLMAN David, 777 S. Mathilda Avenue, #266, Sunnyvale, CA 94087, US,

Legal Representative:

HEILBRUNN Elise R (agent), Beyer Weaver & Thomas L.L.P., 2030 Addison Street, Berkeley, CA 94704, US,

Patent and Priority Information (Country, Number, Date):

Patent: WO 200254235 A2-A3 20020711 (WO 0254235)

Application: WO 2001US46249 20011030 (PCT/WO US2001046249)

Priority Application: US 2000752888 20001228

Designated States: AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CO CR CU

CZ DE DK DM DZ EC EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP

KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PH PL PT RO RU

SD SE SG SI SK SL TJ TM TR TT TZ UA UG UZ VN YU ZA ZW

(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE TR

(OA) BF BJ CF CG CI CM GA GN GQ GW ML MR NE SN TD TG

(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZW

(EA) AM AZ BY KG KZ MD RU TJ TM

Main International Patent Class: G06F-009/45

International Patent Class: G06F-012/02

Publication Language: English

Filing Language: English  
Fulltext Availability:  
Detailed Description  
Claims  
Fulltext Word Count: 5024

English Abstract

Methods and apparatus for identifying objects to enable memory associated with the identified objects to be reclaimed. Methods include identifying one or more objects of a first object type, obtaining one or more addresses of source code adapted for creating the one or more objects identified as the first object type when the source code is executed, and performing class file generation such that the one or more addresses are stored in a data structure in one or more class files.

French Abstract

L'invention concerne des procedes et un appareil permettant d'identifier des objets pour pouvoir recuperer la memoire associee aux objets identifies. Ces procedes consistent a identifier un ou plusieurs objets d'un premier type, a obtenir une ou plusieurs adresses de code source concues pour creer le ou les objets identifies comme etant du premier type lorsque le code source est execute, puis a generer des fichiers de classes de sorte que la ou les adresses soient stockees dans une structure de donnees dans un ou plusieurs fichiers de classes.

Legal Status (Type, Date, Text)

Publication 20020711 A2 Without international search report and to be republished upon receipt of that report.  
Examination 20021017 Request for preliminary examination prior to end of 19th month from priority date  
Search Rpt 20040108 Late publication of international search report  
Republication 20040108 A3 With international search report.

Main International Patent Class: G06F-009/45

Fulltext Availability:

Detailed Description

Detailed Description

... once the local objects are identified, a local table is created at block 204 that includes addresses of all source code (e.g., bytecodes) adapted for creating the local objects during method execution.

The local table may be implemented...

27/5,K/40 (Item 40 from file: 349)  
DIALOG(R) File 349:PCT FULLTEXT  
(c) 2004 WIPO/Univentio. All rts. reserv.

00831815 \*\*Image available\*\*

SYSTEM AND METHOD FOR GENERATING INTERNET SERVICES  
SYSTEME ET PROCEDE D'ELABORATION DE SERVICES INTERNET

Patent Applicant/Assignee:

INNUIITY INC, 1712 Hopkins Crossroads, Minnetonka, MN 55305, US, US  
(Residence), US (Nationality)

Inventor(s):

MCKNIGHT Julie, 909 Wetlyn Court, Farmington, MN 55024, US,  
FROEMING Eric, 230 Central Avenue North #119, Wayzata, MN 55391, US,

Legal Representative:

BRUESS Steven C (agent), Merchant & Gould P.C., P.O. Box 2903,  
Minneapolis, MN 55402-0903, US,

Patent and Priority Information (Country, Number, Date):

Patent: WO 200165399 A2-A3 20010907 (WO 0165399)  
Application: WO 2001US6360 20010228 (PCT/WO US0106360)  
Priority Application: US 2000515064 20000228

Designated States: AE AG AL AM AT (utility model) AT AU AZ BA BB BG BR BY  
BZ CA CH CN CO CR CU CZ (utility model) CZ DE (utility model) DE DK  
(utility model) DK DM DZ EE (utility model) EE ES FI (utility model) FI  
GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV

MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK (utility model)  
SK SL TJ TM TR TT TZ UA UG UZ VN YU ZA ZW  
(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE TR  
(OA) BF BJ CF CG CI CM GA GN GW ML MR NE SN TD TG  
(AP) GH GM KE LS MW MZ SD SL SZ TZ UG ZW  
(EA) AM AZ BY KG KZ MD RU TJ TM

Main International Patent Class: G06F-017/24

International Patent Class: G06F-009/44; G06F-017/30; G06F-017/60;  
G06F-017/21

Publication Language: English

Filing Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 8642

#### English Abstract

A computerized method for automatically generating customized Internet services that are available through an agency. The method comprises providing at least one default service; providing a plurality of attributes; and automatically mapping predetermined attributes selected from the plurality of attributes with the default service, thereby creating a customized Internet service. Alternatively, a computerized method of generating a web page for posting on the Internet. The method comprises providing a page structure; providing a plurality of objects, each object relating to content used to form the web page; and mapping the content with the page structure, thereby creating a web page.

#### French Abstract

L'invention concerne un procede informatique permettant d'elaborer, de maniere automatique, des services Internet personnalises qui sont disponibles par l'intermediaire d'une agence. Ce procede consiste a fournir au moins un service par default ainsi qu'une pluralite d'attributs; a acheminer, de maniere automatique, les attributs predetermines et selectionnes dans la pluralite d'attributs, creant ainsi un service Internet personnalise. Dans un autre mode de realisation, un procede informatique permet d'elaborer une page web destinee a etre publiee sur Internet. Ce procede consiste a etabliir une structure de page et une pluralite d'objets, chaque objet ayant trait au contenu utilise pour creer la page web; et a acheminer le contenu avec la structure de page, creant ainsi une page web.

#### Legal Status (Type, Date, Text)

Publication 20010907 A2 Without international search report and to be republished upon receipt of that report.

Search Rpt 20030206 Late publication of international search report

Republication 20030206 A3 With international search report.

Fulltext Availability:

Claims

#### Claim

... object,  
wherein each web object defines an element located on the web page,  
the web object having source code with embedded object  
placeholders associated with content for the web object; and  
an index for dynamically mapping...  
?t27/5,k/42-43,46-50

27/5,K/42 (Item 42 from file: 349)  
DIALOG(R)File 349:PCT FULLTEXT  
(c) 2004 WIPO/Univentio. All rts. reserv.

00760492 \*\*Image available\*\*

USER CENTRIC PROGRAM PRODUCT DISTRIBUTION

DISTRIBUTION DE PRODUITS PROGRAMMES AXEE SUR L'UTILISATEUR

Patent Applicant/Inventor:

PORTER Swain W, 12511 89th Court, N.E., Kirkland, WA 98034, US, US

(Residence), US (Nationality)

Legal Representative:

AUYEUNG Aloysuis T C, Blakely, Sokoloff, Taylor & Zafman, 7th floor,  
12400 Wilshire Boulevard, Los Angeles, CA 90025-1026, US

Patent and Priority Information (Country, Number, Date):

Patent: WO 200073901 A1 20001207 (WO 0073901)

Application: WO 2000US2605 20000131 (PCT/WO US0002605)

Priority Application: US 99323579 19990601

Designated States: AE AL AM AT AU AZ BA BB BG BR BY CA CH CN CR CU CZ DE DK

DM EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR

LS LT LU LV MA MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ

TM TR TT TZ UA UG UZ VN YU ZA ZW

(EP) AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE

(OA) BF BJ CF CG CI CM GA GN GW ML MR NE SN TD TG

(AP) GH GM KE LS MW SD SL SZ TZ UG ZW

(EA) AM AZ BY KG KZ MD RU TJ TM

Main International Patent Class: G06F-009/44

International Patent Class: G06F-009/445

Publication Language: English

Filing Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 9288

English Abstract

A user centric approach to program product distribution, including a complementary multi-vendor code control system (MVCCS) suitable for use to practice the user centric distribution approach is disclosed. Under the user centric approach, versioning control information of the source/object files of various program products to be installed on various user computer systems are maintained on a user computer system by user computer system basis. Each user computer system or its proxy is provided with a portion or an entire MVCCS to facilitate receipt and storage into a common repository for the user computer system versioning control information of different source/object files of different software vendors, and to facilitate retrieval of selective versions of the different source/object files for the user computer system using versioning control information stored in the common repository for the user computer system. In one embodiment, the MVCCS is further equipped to facilitate receipt and storage into a common library, the different source/object files identified by corresponding universally unique identifiers (UUID), and the versioning control information includes predecessor UUID information. In one embodiment, both the common repository and the common library, as well as the entire MVCCS are disposed on the user computer system.

French Abstract

L'invention concerne une approche axee sur l'utilisateur de la distribution de produits programmes, un systeme de gestion de codes pour plusieurs vendeurs (ou systeme MVCCS) complementaire etant susceptible d'etre utilise pour mettre en oeuvre cette approche de distribution axee sur l'utilisateur. En effet, dans cette approche axee sur l'utilisateur on met a jour, sur un systeme informatique utilisateur et sur la base de plusieurs systemes informatiques utilisateurs, les parametres de controle des versions des fichiers sources/objets des differents produits programmes destines a etre installes sur plusieurs systemes informatiques utilisateurs. Chaque systeme informatique utilisateur, ou son serveur proxy, comprend une partie du systeme MVCCS ou la totalite de celui-ci, ce qui facilite a la fois la reception et le stockage, dans un referentiel commun du systeme informatique utilisateur, des parametres de controle des versions des fichiers sources/objets de plusieurs vendeurs de logiciels, et l'extraction de certaines versions de ces fichiers sources/objets dudit systeme informatique utilisateur utilisant les parametres de controle des versions stockes dans le referentiel commun de ce systeme informatique utilisateur. Dans un mode de realisation, le systeme MVCCS est egalement equipe de maniere a faciliter la reception et le stockage, dans une bibliotheque commune, des fichiers sources/objets

identifies par des identificateurs uniques universels correspondants (ou identificateurs UUID), les parametres de controle des versions incluant notamment des informations UUID precedentes. Enfin, dans un autre mode de realisation, le referentiel commun et la bibliotheque commune, comme l'ensemble du systeme MVCCS, sont installes dans le systeme informatique utilisateur.

Legal Status (Type, Date, Text)

Publication 20001207 A1 With international search report.

Fulltext Availability:

Claims

Claim

... user computer system.

5 The method of claim 1, wherein the method further comprises distributing **source / object files for storage** in a plurality of corresponding libraries, one library

20

for each user computer system, each library **storing source / object files** for a plurality of program product vendors for the corresponding user computer system.

6 The...

...12 The apparatus of claim 8, wherein the code control/distribution system further distributes the **source / object files for storage** in a plurality of corresponding libraries, one library for each user computer system, each library **storing source / object files** for a plurality of program product vendors for the corresponding user computer system.

13 The...

...claim 15, wherein said programming instructions enable the apparatus to be able to distribute the **source / object files for storage** in a plurality of corresponding libraries, one library for each user computer system, each library **storing source / object files** for a plurality of program product vendors for the corresponding user computer system.

23

. The...

27/5,K/43 (Item 43 from file: 349)

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00560512 \*\*Image available\*\*

**STORAGE OF STATIC DATA FOR EFFICIENT ACCESS AND FIELD UPGRADE**

**STOCKAGE DE DONNEES STATIQUES POUR ACCES ET EXTENSION DE ZONE EFFICACES**

Patent Applicant/Assignee:

SOFTBOOK PRESS INC,

Inventor(s):

WALTER Erik,

CONBOY Garth,

DUGA Brady,

Patent and Priority Information (Country, Number, Date):

Patent: WO 200023885 A1 20000427 (WO 0023885)

Application: WO 99US24242 19991015 (PCT/WO US9924242)

Priority Application: US 98173976 19981016

Designated States: AE AL AM AT AU AZ BA BB BG BR BY CA CH CN CR CU CZ DE DK

DM EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR

LS LT LU LV MA MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ

TM TR TT TZ UA UG UZ VN YU ZA ZW GH GM KE LS MW SD SL SZ TZ UG ZW AM AZ

BY KG KZ MD RU TJ TM AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT

SE BF BJ CF CG CI CM GA GN GW ML MR NE SN TD TG

Main International Patent Class: G06F-009/44

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 4561

#### English Abstract

The present invention is a method and apparatus for storing static data in a memory. The static data are represented according to a resource file structure and **included** in a **source code** of an execution code. The source code is compiled to generate a **machine code** which includes the static data and the execution code. The **machine code** is transferred to the memory. The technique allows access to a data field in a data structure embedded in a program code. The data field corresponds to a structure element. A pointer to the data field which is stored in the data structure is obtained. The data field is retrieved using the pointer.

#### French Abstract

L'invention concerne un procede et un dispositif permettant de stocker des donnees statiques dans une memoire, ces donnees etant representees d'apres une structure de fichier des ressources et incorporees a un code source de code d'execution. Le code source fait l'objet d'une compilation, de maniere a fournir un code machine englobant les donnees statiques et le code d'execution. Le code machine est transfere a la memoire. Le procede considere permet d'accéder a une zone dans une structure de donnees integree a un code de programme. La zone correspond a un element de structure. On obtient un pointeur de zone, qui est enregistre dans la structure de donnees. La zone est ensuite recuperee au moyen du pointeur.

#### English Abstract

...in a memory. The static data are represented according to a resource file structure and **included** in a **source code** of an execution code. The source code is compiled to generate a **machine code** which includes the static data and the execution code. The **machine code** is transferred to the memory. The technique allows access to a data field in a...

27/5,K/46 (Item 46 from file: 349)

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00530618 \*\*Image available\*\*

GLOBAL REGISTER SYSTEMS, METHODS, AND COMPUTER PROGRAM PRODUCTS  
SYSTEMES, PROCEDES, ET PRODUITS DE PROGRAMME INFORMATIQUE DE REGISTRES  
GENERAUX

Patent Applicant/Assignee:

SUN MICROSYSTEMS INC,

Inventor(s):

CHESSIN Stephen Alan,

EVANS Rodrick Ison,

WALKER Michael S,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9961970 A2 19991202

Application: WO 99US12063 19990528 (PCT/WO US9912063)

Priority Application: US 9887352 19980529; JP 99146389 19990526

Designated States: AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES

FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU

LV MD MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA

UG UZ VN YU ZW GH GM KE LS MW SD SL SZ UG ZW AM AZ BY KG KZ MD RU TJ TM

AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM

GA GN GW ML MR NE SN TD TG

Main International Patent Class: G06F

Publication Language: English

Fulltext Availability:

Detailed Description  
Claims  
Fulltext Word Count: 5954

#### English Abstract

A system, method and computer program product for compiling and linking a source file and to generate a symbol table associating a global symbol with a register referenced in the source file. The symbol table enables a linker to initialize the global registers using a relocation entry which holds an initializer. The compiler also generates an **object file** from the **source file**. The **object file** includes the global symbol information. A linker links the **object file** potentially with at least one other **object file** or shared library to thereby generate an executable file or shared library. The linker uses the global symbol information contained in the object file to initialize the global registers and to perform relocation operations.

#### French Abstract

L'invention concerne un systeme, un procede, et un produit de programme informatique permettant a la fois de compiler et d'editer un lien a un fichier source, et de produire une table des etiquettes associant un symbole general a un registre consulte dans ledit fichier source. La table des etiquettes permet en outre a un editeur de liens d'initialiser les registres generaux a l'aide d'une entree de translation comportant un initialisateur. Le compilateur produit un fichier objet a partir dudit fichier source, ce fichier objet comprend les informations d'etiquettes generales. Un editeur de liens peut eventuellement lier ce fichier objet a au moins un autre fichier objet ou a au moins une bibliotheque commune, de maniere a produire un fichier ou une bibliotheque commune executable. Cet editeur de liens utilise par ailleurs les informations d'etiquettes generales contenues dans ledit fichier objet pour initialiser les registres generaux et proceder a des operations de translation.

#### English Abstract

...global registers using a relocation entry which holds an initializer. The compiler also generates an **object file** from the **source file**. The **object file** includes the global symbol information. A linker links the **object file** potentially with at least one other **object file** or shared library to thereby generate an executable file or shared library. The linker uses...

27/5,K/47 (Item 47 from file: 349)  
DIALOG(R) File 349:PCT FULLTEXT  
(c) 2004 WIPO/Univentio. All rts. reserv.

00460390

#### METHOD OF RECOVERING SOURCE CODE FROM OBJECT CODE

#### PROCEDE DE RECUPERATION DE CODE SOURCE A PARTIR D'UN CODE OBJET

Patent Applicant/Assignee:

THE SOURCE RECOVERY COMPANY LLC,

Inventor(s):

BRANDES Frederick A,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9850854 A1 19981112

Application: WO 98US9358 19980507 (PCT/WO US9809358)

Priority Application: US 97853029 19970508

Designated States: CN JP AT BE CH CY DE DK ES FI FR GB GR IE IT LU MC NL PT  
SE

Main International Patent Class: G06F-009/45

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 16298

#### English Abstract

A method of recovering source code from object code, comprising

providing a computer program in object code format, disassembling the computer program into assembler code format, including machine instructions and their operands, providing assembler code patterns, and for each such pattern, its equivalent source language command structures, comparing the provided assembler code patterns to be assembler code, to find provided assembler code patterns in the assembler code, and for each such found provided assembler code pattern, assigning to the assembler code portion which makes up the pattern, the equivalent source language command structure.

#### French Abstract

La presente invention concerne un procede de recuperation de code source a partir d'un code objet consistant a obtenir un programme informatique en format de code objet; a desassembler le programme informatique en format de code assembleur comprenant des instructions machine et leurs operandes; a obtenir des modeles de code assembleur et pour chacun des modeles, les structures de commande equivalentes en langage source; a comparer des modeles de code assembleur obtenus au code assembleur de facon a trouver des modeles de code assembleur obtenus dans le code assembleur; et pour chaque modele de code assembleur obtenu ainsi trouve a attribuer la structure de commande de langage source equivalente a la partie de code assembleur qui constitue le modele.

Main International Patent Class: G06F-009/45

Fulltext Availability:

Claims

#### Claim

... of claim 14 in which the step of creating a data portion of the recovered source code includes an analysis of the types of machine instructions that employ said operands.  
SUBSTITUTE SHEET (RULE 26)

16 A method of recovering source code...

27/5,K/48 (Item 48 from file: 349)

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00415572 \*\*Image available\*\*

EMBEDDED WEB SERVER

SERVEUR WEB INTEGRE

Patent Applicant/Assignee:

AGRANAT SYSTEMS INC,

Inventor(s):

AGRANAT Ian D,

GIUSTI Kenneth A,

LAWRENCE Scott D,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9806033 A1 19980212

Application: WO 97US13817 19970808 (PCT/WO US9713817)

Priority Application: US 9623373 19960808

Designated States: JP AT BE CH DE DK ES FI FR GB GR IE IT LU MC NL PT SE

Main International Patent Class: G06F-009/44

International Patent Class: G06F-17:30

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 55647

#### English Abstract

An embedded graphical user interface employs a World-Wide-Web communications and display paradigm. The development environment includes an HTML compiler which recognizes and processes a number of unique extensions to HTML. The HTML compiler produces an output which is in the source code language of an application to which the graphical user



interface applies. A corresponding run-time environment includes a server which serves the compiled HTML documents to a browser.

#### French Abstract

Une interface graphique utilisateur integree utilise un paradigme de communication et d'affichage Web. L'environnement de developpement comprend un compilateur en langage de balisage hypertexte (HTML), qui reconnait et traite un certain nombre d'extensions uniques du HTML. Le compilateur HTML produit une sortie dans le langage du code source d'une application utilisant l'interface graphique utilisateur. Un environnement charge a l'execution correspondant comprend un serveur qui transmet les documents HTML compiles a un explorateur Web.

#### Fulltext Availability:

Claims

#### Claim

... of the mark-up language, the compiler producing as an output a representation in the **native application source code language** of the document, **including** a copy of the source code fragment.

15 The apparatus of claim 14, wherein the...

27/5,K/49 (Item 49 from file: 349)

DIALOG(R) File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00307152 \*\*Image available\*\*

OPTIMIZING TIME AND TESTING OF HIGHER LEVEL LANGUAGE PROGRAMS

OPTIMISATION DU TEMPS ET DES ESSAIS DE PROGRAMMES DE LANGAGES DE NIVEAU SUPERIEUR

Patent Applicant/Assignee:

GREEN HILLS SOFTWARE INC,

O'DOWD Daniel D,

KLEIDERMACHER David N,

Inventor(s):

O'DOWD Daniel D,

KLEIDERMACHER David N,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9525304 A1 19950921

Application: WO 95US3003 19950314 (PCT/WO US9503003)

Priority Application: US 94212600 19940314

Designated States: AU CA ES JP MX PL RU UA AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE

Main International Patent Class: G06F-011/34

International Patent Class: G06F-11:30; G06F-09:455

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 16943

#### English Abstract

A method for time use analysis of a higher level language program is performed by displaying source code lines (56) in descending order according to the amount of time spent by the program to execute machine code (94) into which the source code lines have been compiled. Source code lines are displayed (96) arranged in order according to the percentages of the amounts of time spent in execution during runs of the program. A digital processing apparatus (10) for performing the analysis includes a display (15) for showing the source code lines (56) that require the most time of execution, a selection apparatus (108) for selecting those source code lines having the greater opportunity for significant corrective action, and displaying the various selected source code lines (104) in the order in which the lines are kept in the program along with the corresponding time spent by the program to execute machine code.

## French Abstract

L'invention concerne un procede pour analyser l'utilisation du temps d'un programme de langages de niveau superieur. Ce procede consiste a visualiser des lignes de code source (56) en ordre decroissant selon le temps passe par le programme pour executer le code machine (94) dans lequel les lignes de code source ont ete compilees. Les lignes de code source sont visualisees (96) en etant classees en fonction des pourcentages de temps consacre a l'execution pendant le passage du programme. L'invention concerne egalement un appareil de traitement numerique (10) pour executer l'analyse, qui comprend un dispositif de visualisation (15) pour montrer les lignes de code source (56), necessitant le plus de temps de fonctionnement; un appareil de selection (108) pour selectionner ces lignes de code source les plus susceptibles d'assurer une action corrective significative, et pour afficher les diverses lignes de code source (104) selectionnees dans l'ordre dans lequel elles sont conservees dans le programme avec le temps correspondant consacre par le programme a l'execution du code machine.

## Fulltext Availability:

Claims

## Claim

... line.

41 The digital processing apparatus of Claim 39 wherein in said displaying means, said **source code** line indicia includes at least some of the text of the **machine instructions** into which said source code line has been translated,

42 The digital processing apparatus of...

27/5,K/50 (Item 50 from file: 349)  
DIALOG(R)File 349:PCT FULLTEXT  
(c) 2004 WIPO/Univentio. All rts. reserv.

00279043

**METHOD AND APPARATUS FOR VECTORIZING THE CONTENTS OF A READ ONLY MEMORY DEVICE WITHOUT MODIFYING UNDERLYING SOURCE CODE**  
**PROCEDE ET APPAREIL POUR VECTORISER LE CONTENU D'UN DISPOSITIF A MEMOIRE ROM SANS MODIFIER LE CODE SOURCE SOUS-JACENT**

Patent Applicant/Assignee:

APPLE COMPUTER INC,

Inventor(s):

WETMORE Russ,

NGUYEN Philip,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9427220 A1 19941124

Appication: WO 94US4994 19940506 (PCT/WO US9404994)

Priority Application: US 9358876 19930506

Designated States: AT AU BB BG BR BY CA CH CN CZ DE DK ES FI GB GE HU JP KG  
KP KR KZ LK LU LV MD MG MN MW NL NO NZ PL PT RO RU SD SE SI SK TJ TT UA  
UZ VN AT BE CH DE DK ES FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM  
GA GN ML MR NE SN TD TG

Main International Patent Class: G06F-009/45

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 7124

## English Abstract

A method and apparatus for generating an object file that facilitates patching and the introduction of new function. The present invention accomplishes this without disturbing the original source file. The present invention is particularly useful in the generation of programs that will exist on a static device such as a Read Only Memory (ROM) device. The present invention requires that access to routines in the

object file be referenced through a vector table located in Random Access Memory (RAM). If a routine in ROM must be patched (i.e. replaced) or if new function is added, the vector table is modified. Modification may be either changing the contents of an existing entry (replacement) or adding a new entry (new function). Generally, this modification involves the steps of: identifying the entry points in the object file to create a vector source table; generating a vector object table from the vector source table; generating a symbol table from the vector object table; comparing entry points in the object files to entries in the symbol table; when a match is found, modifying the entry point of the object file to reference a corresponding entry in the vector table. Since only the object file is modified, the original source file is not disturbed.

#### French Abstract

L'invention se rapporte a un procede et a un appareil servant a generer un fichier objet qui facilite les operations de modification et l'introduction d'une nouvelle fonction, sans perturber le fichier source original. Cette invention est particulierement utile pour generer des programmes qui resident sur un dispositif statique tel qu'un dispositif a memoire morte (memoire ROM). A cet effet, il faut que l'accès aux sous-programmes du fichier objet se fasse par renvoi au moyen d'une table de vecteurs se trouvant dans une memoire a acces selectif (memoire RAM). Lorsqu'il s'agit de modifier (c'est-a-dire remplacer) un sous-programme dans la memoire ROM ou lorsqu'il s'agit d'ajouter une nouvelle fonction, la table vectorielle est modifiée. La modification peut etre soit un changement du contenu d'une entree existante (remplacement) soit l'adjonction d'une nouvelle entree (nouvelle fonction). Generalement, cette modification comporte les etapes suivantes: identifier les points d'entree dans le fichier objet, afin de creer une table vectorielle source; former une table vectorielle objet a partir de la table vectorielle source; former une table de symboles a partir de la table vectorielle objet; comparer les points d'entree dans les fichiers objets avec les entrees dans la table de symboles; et, lorsqu'une correspondance est constatée, modifier le point d'entree du fichier objet afin de creer un renvoi a une entree correspondante dans la table vectorielle. Etant donne que seul le fichier objet est modifié, le fichier source original n'est pas perturbe.

Main International Patent Class: G06F-009/45

Fulltext Availability:

Claims

#### Claim

... table  
object file; and  
f) generating a vector table initialization file from said vector  
table source file; and  
g) appending said vector table initialization file to said vectorized  
object file .

2 The method as recited in Claim 1 wherein said vector table object  
file is...

?t27/5,k/54

27/5,K/54 (Item 54 from file: 349)

DIALOG(R) File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00186448 \*\*Image available\*\*

OPERATING SYSTEM AND DATA BASE

SYSTEME D'EXPLOITATION ET BASE DE DONNEES

Patent Applicant/Assignee:

AMDAHL CORPORATION,  
KNUDSEN Helge,  
CHONG Daniel T,  
YAFFE John,  
TAUGHER James E,  
ROBERTSON Michael,

PLAZAK Zbigniew,  
Inventor(s):  
KNUDSEN Helge,  
CHONG Daniel T,  
YAFFE John,  
TAUGHER James E,  
ROBERTSON Michael,  
PLAZAK Zbigniew,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9103791 A1 19910321  
Application: WO 90US5007 19900904 (PCT/WO US9005007)  
Priority Application: US 89862 19890901; US 89298 19891213  
Designated States: AT AU BE CA CH DE DK ES FR GB IT JP LU NL SE US US  
Main International Patent Class: G06F-015/40  
Publication Language: English  
Fulltext Availability:  
Detailed Description  
Claims  
Fulltext Word Count: 27700

English Abstract

A system for program development and execution consisting of a high level programming language based on a four part rule organization (31), consisting of a rule definition, a list of conditions, a list of actions taken upon satisfaction of a corresponding condition, and a list of exception handlers. Data access events are supplied through a table access method (21) which provides an interface to the variety of sources of data (22-30) coupled to the system. A table data store (103) organizes data in an object oriented, relational system, where each table is ordered on a primary key (502). The table access method (21) performs selection and ordering operations on the tables accessible through the table access method (21), implements and triggers invalidation routines upon data access events, and provides a common view of data stored across heterogeneous data stores coupled through servers (203-205) to the table access method (21). Ordered tables (103) are subdividable by additional parameters associated with table names.

French Abstract

Système pour le développement et l'exécution de programmes comprenant un langage de programmation évoluée à base d'une organisation de règles en quatre parties (31). Ledit système comprend une définition de règles, une liste de conditions, une liste d'actions effectuées lorsqu'une condition correspondante est satisfaite, et une liste de programmes de gestion d'exceptions. Des événements d'accès aux données sont déterminés grâce à une méthode d'accès à tables (21), qui crée une interface aux diverses sources de données (22-30) couplées au système. Une mémoire de tables de données (103) organise des données dans un système relationnel travaillant au niveau des objets, dans lequel chaque table est ordonnée sur un indicatif majeur (502). La méthode d'accès à tables (21) effectue des opérations de sélection et d'ordonnance sur les tables rendues accessibles par ladite méthode d'accès à tables (21). Ladite méthode met en oeuvre et déclenche également des sous-programmes d'invalidation lors d'événements d'accès aux données, et établit une vue commune des données stockées dans des mémoires de données hétérogènes couplées par des serveurs (203-205) à la méthode d'accès à tables (21). On peut subdiviser des tables ordonnées (103) utilisant des paramètres additionnels associés aux noms des tables.

Fulltext Availability:  
Claims

Claim

... said programmer;  
storing said first source-coded rule into said tables, by  
said means for storing ;  
translating said first source - coded rule into a first  
object - coded rule, by said translator means;  
storing said first object-coded rule into said tables, by...

...storing said second source-coded rule, as changed, into said  
tables, by said means for storing ;  
translating said second source - coded rule into a second  
object - coded rule, by said translator means;  
storing said second object-coded rule into said tables, by...

1/5,k

1/5,k/1

DIALOG(R)File 349:PCT FULLTEXT

(c) 2004 WIPO/Univentio. All rts. reserv.

00476830 \*\*Image available\*\*

**METHOD AND APPARATUS FOR STATIC AND DYNAMIC GENERATION OF INFORMATION ON A  
USER INTERFACE****PROCEDE ET DISPOSITIF DE GENERATION STATIQUE ET DYNAMIQUE D'INFORMATION SUR  
UNE INTERFACE D'UTILISATEUR**

Patent Applicant/Assignee:

LUTRIS TECHNOLOGIES INC,

Inventor(s):

MORGAN Paul A,

DIEKHANS Mark E,

CLARK Kyle,

Patent and Priority Information (Country, Number, Date):

Patent: WO 9908182 A1 19990218

Application: WO 98US16348 19980805 (PCT/WO US9816348)

Priority Application: US 9754817 19970805

Designated States: AL AM AT AU AZ BA BB BG BR BY CA CH CN CU CZ DE DK EE ES

FI GB GE GH GM HR HU ID IL IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MD

MG MK MN MW MX NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT UA UG UZ

VN YU ZW GH GM KE LS MW SD SZ UG ZW AM AZ BY KG KZ MD RU TJ TM AT BE CH

CY DE DK ES FI FR GB GR IE IT LU MC NL PT SE BF BJ CF CG CI CM GA GN GW

ML MR NE SN TD TG

Main International Patent Class: G06F-007/00

International Patent Class: G06K-007/10

Publication Language: English

Fulltext Availability:

Detailed Description

Claims

Fulltext Word Count: 7438

**English Abstract**

Methods and systems of the present invention (100) enable a user to request a web page with static information and dynamic information by invoking a presentation object located on a server (101). The presentation object is created by compiling object-oriented code (121) together with structured text (116), such as text formatted using HTML. The structured text can be used to display static information on the web page. People can modify the structured text portion of the presentation object (124) without modifying the object-oriented programming language. More sophisticated software developers can use the object-oriented programming language to generate information to be used in the web page. By combining object-oriented code and structured text, users can develop complex web pages for use in enterprise networks.

**French Abstract**

Procedes et systemes (100) permettant a un utilisateur de demander a une page Web une information statique et une information dynamique, ce qui consiste a invoquer un objet de presentation localise sur un serveur (101). On cree cet objet de presentation par compilation d'un code oriente objet (121) et d'un texte structure (116), tel qu'un texte formate au moyen de HTML. On peut utiliser ce texte structure afin d'afficher une information statique sur la page Web. On peut modifier la partie de texte structure de l'objet de presentation (124) sans modifier le langage de programmation oriente objet. Des createurs de logiciels plus sophistiques peuvent mettre en application le langage de programmation oriente objet afin de generer l'information a utiliser dans la page Web. La combinaison d'un code oriente objet et d'un texte structure permet aux utilisateurs d'elaborer des pages Web complexes afin de les mettre en application dans des reseaux d'entreprise.

Fulltext Availability:

Detailed Description

## Claims

### Detailed Description

... text

source file;

FIG. 4 is a flowchart of the steps associated with converting an **object -text source**

**file** into an **integrated** object-oriented **code** module; and

FIG. 5 is a flowchart diagram indicating the steps associated with invoking a...displays "Unknown user."

FIG. 4 is a flowchart of the steps associated with converting an **object -text source file** 121 into an **integrated** object-oriented **code** module. A compiler performing this conversion step substitutes static structured text with object-oriented code...

### Claim

... of displaying static information and dynamically generated information on a user interface

comprising:

receiving an **object -text source file** that **integrates** structured text and object

oriented **code** ;

converting the structured text and object-oriented code into an integrated object

oriented code module...

...displaying static information

and dynamic information on a user interface comprising:

a memory having an **object -text source file** that **combines** structured text and

object-oriented **code** ;

a processor that executes instructions to convert the structured text and object-oriented **code** in the **object -text source file** into an **integrated** object-oriented **code** module, and compile the integrated object-oriented **code** module into the presentation object.

12 The system in claim ...capable of displaying static information and dynamic information on a user interface, by: receiving an **object -text source file** that **integrates** structured text and object oriented **code** ;

converting the structured text and object-oriented code into an integrated object

oriented code module...

?

[Web](#) [Images](#) [Groups](#) [News](#) [more »](#)[Advanced Search](#)  
[Preferences](#)**Web**Results 1 - 10 of about 1,090,000 for **object file source text**. (0.68 seconds)[visual basic, vb, active server pages \(ASP\), java, javascript, c, c++ ...](#)... Explains the concepts of Garbage Collection and **object** resurrection in ... AM Language:[.Net Convert a Binary-File to XML ... site...a PRO version of Planet Source Code ...](#)<http://www.planet-source-code.com/> - 37k - [Cached](#) - [Similar pages](#)**! Aware: Man pages: Object File Utilities**... size(1) - display **object file** segment sizes (**text**, data and bss) {oss} Manpages: FreeBSD RedHat Solaris NetBSD **Source** code: OpenBSD FreeBSD. ...<http://www.rocketaware.com/man/spec/softdev/objtool/> - 14k - [Cached](#) - [Similar pages](#)**GOLDParser Object**... must perform the following tasks to use the GOLDParser **object**. Read a previously developed Compiled Grammar Table **file**; Open the **source** of the **text** to be parsed ...<http://www.devincook.com/goldparser/doc/engine/activex/object-goldparser.htm> - 12k - [Cached](#) - [Similar pages](#)**File: rimport\_cmd.rb**... to the ri distribution, so you may want to dig into that **source** to see ... At the end of each **file** is a line of code that marshals the **object** and writes it ...[http://rimport.rubyforge.org/doc/files/rimport\\_cmd\\_rb.html](http://rimport.rubyforge.org/doc/files/rimport_cmd_rb.html) - 10k - [Cached](#) - [Similar pages](#)**GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999 ...**... material from a header **file** that is part of the Library, the **object** code for the work may be a derivative work of the Library even though the **source** code is not ...<http://www.gnu.org/copyleft/lesser.txt> - 27k - [Cached](#) - [Similar pages](#)**The Interactive disassembler - OMF files**... An **object file** to assembly language **file** conversion utility Only the **source text** is provided. Authors: Robert F. Day, Robin Hilliard Dated: May 1992. OBJLIB.ZIP ...<http://www.rosprobank.ru/~ig/obj/> - 4k - [Cached](#) - [Similar pages](#)**WhatIs.com**... ASI, Borland C++/Turbo C Assembler Include **file**. ASM, Assembler Language**source file**. ASM, Pro/E assembly **file**. ASO, Astound Dynamite **Object**. ...<http://www.whatis.com/fileFormatA/0,289933,sid9,00.html> - 55k - [Cached](#) - [Similar pages](#)**Mason HQ: Compile Mason component source (version 1.25)**... The sub is called with a single parameter, a scalar reference to the **text** portion of the ... are in terms of the **source file** instead of the **object file**. ...<http://www.masonhq.com/docs/manual/Compiler.html> - 15k - Mar 8, 2004 - [Cached](#) - [Similar pages](#)**Take advantage of the Tabular Data Control data source object in ...**... Listing B shows a page that uses the Tabular Data Control data **source object** to display the delimited **text file** from Listing A. The combined result of these ...<http://builder.com.com/5100-6371-1058715.html> - 35k - [Cached](#) - [Similar pages](#)**Object-Oriented Software in Ada 95 2nd edition****Object-Oriented Software in Ada 95 2nd edition**, ... which is removed before the line is written to the **file**. The **source** code held in this archive format for the all ...[http://www.brighton.ac.uk/ada95/extra/home\\_p.html](http://www.brighton.ac.uk/ada95/extra/home_p.html) - 7k - [Cached](#) - [Similar pages](#)Search for **object file source text** on [Images](#), [Groups](#), [News](#)





US Patent &amp; Trademark Office

[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☐ The ACM Digital Library ☒ The Guide


**THE GUIDE TO COMPUTING LITERATURE**
[Feedback](#) [Report a problem](#) [Satisfaction survey](#)
Terms used **embedded source object file**Found **60,668** of **792,123**

Sort results by

☒ [Save results to a Binder](#)
Try an [Advanced Search](#)

Display results

☐ [Search Tips](#)
Try this search in [The Digital Library](#)
☐ Open results in a new window

Results 1 - 20 of 200

Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

Relevance scale ☐ ☐ ☐ ☐ ☐
**1 [Improved interpretation of UNIX-like file names embedded in data](#)**

Douglas W. Jones

August 1984 **Communications of the ACM**, Volume 27 Issue 8Full text available: [pdf\(427.31 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

When the data processed by a program span several files, the common practice of including file names as data in some of the files leads to difficulties in moving or sharing that data. In systems using tree structured directories, this problem can be solved by making a syntactic distinction between absolute and relative file names.

**Keywords:** UNIX, directory management, file name interpretation, file systems, macro parameters, programming environments, scope rules, text insertion

**2 [Word segmentation and recognition for web document framework](#)**

Chi-Hung Chi, Chen Ding, Andrew Lim

November 1999 **Proceedings of the eighth international conference on Information and knowledge management**Full text available: [pdf\(1.26 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

It is observed that a better approach to Web information understanding is to base on its document framework, which is mainly consisted of (i) the title and the URL name of the page, (ii) the titles and the URL names of the Web pages that it points to, (iii) the alternative information source for the embedded Web objects, and (iv) its linkage to other Web pages of the same document. Investigation reveals that a high percentage of words inside the document framework are "compound words& ...

**3 [Practical use of a polymorphic applicative language](#)**

Butler W. Lampson, Eric E. Schmidt

January 1983 **Proceedings of the 10th ACM SIGACT-SIGPLAN symposium on Principles of programming languages**Full text available: [pdf\(1.84 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Assembling a large system from its component elements is not a simple task. An adequate notation for specifying this task must reflect the system structure, accommodate many configurations of the system and many versions as it develops, and be a suitable input to the many tools that support software development. The language described here applies the ideas of  $\lambda$ -abstraction, hierarchical naming and type-checking to this problem. Some preliminary experience with its use is also given.

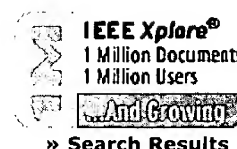
**4 [The design and implementation of an object-oriented toolkit for 3D graphics and](#)**



# IEEE Xplore®

RELEASE 1.6

Welcome  
United States Patent and Trademark Office


[Help](#) [FAQ](#) [Terms](#) [IEEE Peer Review](#)
[Quick Links](#)

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

Your search matched **4** of **1011253** documents.

A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.

### Refine This Search:

You may refine your search by editing the current search expression or entering a new one in the text box.



☐ Check to search within this result set

### Results Key:

**JNL** = Journal or Magazine   **CNF** = Conference   **STD** = Standard

#### 1 Extending software quality assessment techniques to Java systems

*Patenaude, J.-F.; Merlo, E.; Dagenais, M.; Lague, B.;*

Program Comprehension, 1999. Proceedings. Seventh International Workshop on , 5-7 May 1999

Pages:49 - 56

[\[Abstract\]](#)   [\[PDF Full-Text \(88 KB\)\]](#)   **IEEE CNF**

#### 2 In methods we trust?

*Hohmann, L.;*

Computer , Volume: 30 , Issue: 10 , Oct. 1997

Pages:119 - 121

[\[Abstract\]](#)   [\[PDF Full-Text \(236 KB\)\]](#)   **IEEE JNL**

#### 3 Ephemeral Java source code

*Eisenbach, S.; Sadler, C.;*

Distributed Computing Systems, 1999. Proceedings. 7th IEEE Workshop on Future Trends of , 20-22 Dec. 1999

Pages:9 - 14

[\[Abstract\]](#)   [\[PDF Full-Text \(400 KB\)\]](#)   **IEEE CNF**

#### 4 Monitoring distributed embedded systems

*Ford, R.;*

Applied Computing, 1990., Proceedings of the 1990 Symposium on , 5-6 April 1990

Pages:237 - 244

[\[Abstract\]](#)   [\[PDF Full-Text \(716 KB\)\]](#)   **IEEE CNF**

[Web](#) [Images](#) [Groups](#) [News](#) [more »](#)

line version control

Search

[Advanced Search](#)  
[Preferences](#)**Web**Results 1 - 10 of about 3,430,000 for **line version control**. (0.44 seconds)**Version Control Software**

Sponsored Link

[www.seapine.com](http://www.seapine.com) Surround SCM -Branching & Merging, Checkin/out, **version** history & more**Version Control with Subversion**

Sponsored Links

... **Version Control** with Subversion, a free book about Subversion, a new **version control** system designed ... For now, it is simply a place to read the on-line HTML and ...<http://svnbook.red-bean.com/> - 3k - Mar 8, 2004 - [Cached](#) - [Similar pages](#)**TLIB Version Control for Windows - Pricing**... It includes: TLIB **Version Control** for Windows (both 16-bit and 32-bit GUI **versions**);, a Win32 console (command-line) **version** of TLIB (Intel-architecture CPUs ...<http://www.burtonsys.com/old/html/prices.html> - 7k - [Cached](#) - [Similar pages](#)**Control Line Modeling**

Complete source of quality airplane model kits and accessories.

[BJM-Home.com](http://BJM-Home.com)**Version Control Software**PVCS **Version Manager** for software development teams  
[www.merant.com](http://www.merant.com)**Version Control Tools**

SourceSafe, Perforce, and CVS tools Analysis, diffs, and integrations

<http://CodeHistorian.com>**Version control-Compare**Compare prices, tax, shipping, & store ratings for Version Control.  
[www.nextag.com](http://www.nextag.com)**Bitsafe**Bitsafe is a **version control** system for Windows and Unix developers  
[www.bitmanager.de/bitsafe](http://www.bitmanager.de/bitsafe)[See your message here...](#)**TLIB Version Control for Windows compares and shows the ...**... use the included OS/2 command-line **version** of TLIB ... three common kinds of end-of-line delimiters: LF ... Configurable **control** over which text format is generated by ...<http://www.hallogram.com/tlibversion/> - 9k - [Cached](#) - [Similar pages](#)**CVS Command-line Basics**... and developers accustomed to IDE-integrated change-**control** tools, the CVS command **line** can appear ... To fetch the latest **version** from the CVS repository: ...  
<http://titanium.dstc.edu.au/version-control/cvs-cmd-basics.shtml> - 34k - Mar 8, 2004 -[Cached](#) - [Similar pages](#)**Linux Version Control & Configuration Management Tools**... Standards fans take note: the SCCS command **line** is standardized in the Single Unix Specification, **Version 2**. Commercial **Version Control** Products. ...<http://linas.org/linux/cmvc.html> - 14k - [Cached](#) - [Similar pages](#)**Version Control Systems Comparison**... Changes are file-specific. Tracking **Line-wise** File History. Does the **version control** system has an option to track the history of the file **line-by-line**? ...<http://better-scm.berlios.de/comparison/comparison.html> - 42k - Mar 8, 2004 - [Cached](#) - [Similar pages](#)**Version Control**... text files, but binary files as well, so you can integrate **version control** for all ... with HomeSite and Studio, it also has a simple command-line interface which ...<http://hshelp.com/rcs.html> - 17k - [Cached](#) - [Similar pages](#)**Version control glossary**... A shared database with the complete revision history of all files under **version control**.... Someone must go through the file **line by line** to accept one set of ...[https://www.helixcommunity.org/nonav/docs/ddCVS\\_cvsglossary.html](https://www.helixcommunity.org/nonav/docs/ddCVS_cvsglossary.html) - 9k - [Cached](#) -

[Web](#) [Images](#) [Groups](#) [News](#) [more »](#)

line linking compiler

[Advanced Search](#)  
[Preferences](#)**Web**Results 1 - 10 of about 90,600 for line linking compiler. (0.24 seconds)**Re: linking** a shared library with -pthread omits -pthread on the ...... CC -**compiler**="\$2" +**compiler**=\$CC\_LT\_AC\_TAGVAR(**compiler**, \$1)=\$CC ... **Re: linking** a shared library with -pthread omits -pthread on the link **line**., Albert Chin ...<http://mail.gnu.org/archive/html/libtool-patches/2002-12/msg00012.html> - 13k - [Cached](#) - [Similar pages](#)**Re: linking** a shared library with -pthread omits -pthread on ...... Subject: **Re: linking** a shared library with -pthread omits -pthread on the link **line**. ... that libtool will be executed with the identical **compiler** and options ...<http://mail.gnu.org/archive/html/libtool-patches/2002-12/msg00016.html> - 15k - [Cached](#) - [Similar pages](#)[ [More results from mail.gnu.org](#) ]**Examples Compiler Results**Examples **Compiler** Results. ... Function should return a value in function Main(int,char \* \*) : } **Linking**... airport.c: Warning on **line** 330: Function should return a ...<http://www.cs.vu.nl/pub/eliens/jelle/excompil.html> - 13k - [Cached](#) - [Similar pages](#)**Digital Mars - D Compiler**... The actual **linking** is done by running gcc ... The D **compiler** dmd uses the following environment variables: DFLAGS The ... if it were appended to the command **line** to dmd ...<http://www.digitalmars.com/d/alpha.html> - 14k - [Cached](#) - [Similar pages](#)**WebHostingTalk**... Just one thing: How do I do **linking** using this **compiler**? In fact, how does **linking** work in general? ... not so many people can help you with command **line** options. ...<http://www.webhostingtalk.com/archive/thread/84298-1.html> - 6k - [Cached](#) - [Similar pages](#)**Programming in standard C and C++**... Instantiation via the command **line** Single files ... Libraries More on **linking** template code ... programming Other implementation dependencies **Compiler** diagnostics C ...[http://uw713doc.sco.com/en/SDK\\_cprog/CONTENTS.html](http://uw713doc.sco.com/en/SDK_cprog/CONTENTS.html) - 68k - [Cached](#) - [Similar pages](#)**C/C++ Building Reference (Visual C++ Concepts)**... setting **compiler** options in the development environment or on the command **line**.**Compiler** Options Provides **links** to topics discussing using **compiler** options. ...[http://msdn.microsoft.com/library/en-us/vccore/html/\\_core\\_overviews.3a\\_.compiling\\_and\\_linking.asp](http://msdn.microsoft.com/library/en-us/vccore/html/_core_overviews.3a_.compiling_and_linking.asp) - 15k - [Cached](#) - [Similar pages](#)**Linkers -- Indiana University**... **Linking** can be suppressed with **compiler** options. ... o .) Although the linker may be invoked using the ld command from the command **line**, makefile or ...<http://www.indiana.edu/~rac/hpc/pl/linkers.html> - 11k - Mar 9, 2004 - [Cached](#) - [Similar pages](#)**PGHPF Compiler User's Guide - 1 Getting Started**... The driver sets the HPF **compiler** and Fortran **compiler** switches and assembles and **links** the program. It lets you pass command-line options to any of the various ...[http://www.pgroup.com/ppro\\_docs/pghpf Ug/hpfug03.htm](http://www.pgroup.com/ppro_docs/pghpf Ug/hpfug03.htm) - 18k - [Cached](#) - [Similar pages](#)**ocamlc**... as ocamlc, but compiled with the native-code **compiler** ocamlpt(1 ... the object files (.cmo files) given on the command **line**, instead of **linking** them into ...<http://ccrma-www.stanford.edu/planetccrma/man/man1/ocamlc.1.html> - 9k - [Cached](#) - [Similar pages](#)


[Web](#) [Images](#) [Groups](#) [News](#) [more »](#)


[Advanced Search](#)  
[Preferences](#)

## Web

 Results 1 - 10 of about **111,000** for **source line linking compiler**. (0.34 seconds)

### Programming Tools Guide

... for run-time compatibility Dynamic **linking** programming interface ... Named files Passing command **line** arguments C ... behavior Phases of translation **Source** files and ...

<http://docsrv.sco.com:507/en/tools/CONTENTS.html> - 101k - [Cached](#) - [Similar pages](#)

### Digital Mars - D Compiler

... in the output directory -op normally the path for .d **source** files is ... The actual **linking** is done by running gcc ... as if it were appended to the command **line** to dmd ...

<http://www.digitalmars.com/d/alpha.html> - 14k - [Cached](#) - [Similar pages](#)

### Linking VF subroutine to VC++ .NET **source** - Intel® Fortran ...

... The last **line** of the error list says -. ... Thanks,. Kevin. Reply, Re: **Linking** VF subroutine to VC++ .NET **source**, Options, Mark Message as New, ...

<http://softwareforums.intel.com/ids/board/message?board.id=5&message.id=7805> - 101k - [Cached](#) - [Similar pages](#)

### Fortran User's Guide: Contents

... Invoking the **Compiler** Compile-Link Sequence Command-**Line** File Name Conventions **Source** Files **Source** File Preprocessors Separate Compiling and **Linking** Consistent ...

[http://www.ictp.trieste.it/~manuals/programming/sun/fortran/user\\_guide/](http://www.ictp.trieste.it/~manuals/programming/sun/fortran/user_guide/) - 9k - [Cached](#) - [Similar pages](#)

### Salford C/C++ **Compiler**

... also provides you with dynamic **linking** of libraries ... The designers of the **compiler** recognised that one ... a diagnostic pinpointing the relevant **source line** on the ...

<http://www.qtsoftware.de/salford/products/scc.html> - 11k - [Cached](#) - [Similar pages](#)

### Using **Links** With **Compiler** Errors

... there are errors, when the **compiler** finishes the ... Insight will automatically setup the **source links** and run ... error message and the erroneous **source line** will be ...

<http://www.sourcedyn.com/docs35/ad918949.htm> - 4k - [Cached](#) - [Similar pages](#)

### C/C++ Building Reference (Visual C++ Concepts)

... from a command prompt using command-**line** tools. ... **source** files using the Visual C++ **source** editor or a ... **Linking** Describes the linker, which combines code from the ...

[http://msdn.microsoft.com/library/en-us/vccore/html/\\_core\\_overviews.3a\\_.compiling\\_and\\_linking.asp](http://msdn.microsoft.com/library/en-us/vccore/html/_core_overviews.3a_.compiling_and_linking.asp) - 15k - [Cached](#) - [Similar pages](#)

### User's Guide - Chapter 4. Editing, Compiling, **Linking**, and Running ...

... Command **Line**; Specifying Options in the **Source** File; Passing Command-**Line** Options to ... C Preprocessor; Avoiding Preprocessing Problems; **Linking** XL Fortran ...

[http://hpcf.nersc.gov/vendor\\_docs/ibm/xlf/html/UG26.HTM](http://hpcf.nersc.gov/vendor_docs/ibm/xlf/html/UG26.HTM) - 8k - [Cached](#) - [Similar pages](#)

### keywords" content="create **source** code, edit prepared C **source** code ...

... is in C. If you leave the .CPP extension the **compiler** will assume ... **Linking** in Libraries. ... fact that the first non-white space character on the **source line** is a ...

<http://www.geocities.com/learnprogramming123/Clesson1.htm> - 26k - [Cached](#) - [Similar pages](#)

### PGHPF **Compiler** User's Guide - 1 Getting Started

... preprocessor #include directive from within HPF **source** files (includes ... a .F extension or the -Mpreprocess command **line** argument).When **linking** a program ...

[http://www.pgroup.com/ppro\\_docs/pghpf Ug/hpfug03.htm](http://www.pgroup.com/ppro_docs/pghpf Ug/hpfug03.htm) - 18k - [Cached](#) - [Similar pages](#)

# The GNU G++ Compiler

There are many different C and C++ compilers for UNIX environments. Most of these are specific for a particular architecture and operating system. The fine folks in the GNU project have created a C/C++ compiler that has been ported to many different architectures. The name of the compiler is `gcc` or `g++` for the C or C++ compiler respectively. Because of the similarities between C and C++, the two commands actually call the same compiler with different default options. Because of the similarity, the rest of this document will talk exclusively about `g++`. The discussion will automatically apply to `gcc` unless stated otherwise.

Creating an executable program is a two step process. First the source code is *compiled* to object code. The object code may or may not be stored as a separate file, depending on the compiler used and the compiler options. Second, the object code is *linked* with other collections of object code (including system libraries) to form an executable file.

The GNU `g++` compiler can be used to control both the compilation phase and the linking phase of creating executable files.

## Options

Like most UNIX commands, `g++` has numerous options. The most useful of these are

- v Prints out the version of the compiler. This option is usually used by itself.
- c Compiles source code files to object files and stops. The default behavior of the compiler is to compile the given source code files and link them directly into an executable file. The `-c` option is useful for efficiently making projects that have multiple source code files.
- o filename  
Specifies the name of the output file. Without this option, executable files have the default name of `a.out`. The default name of an object file is the same as a source code file, but with a `.o` extension replacing whatever source code extension existed.
- O Turns on optimization. Allows the compiler to modify the code as it's compiling and linking to produce smaller and/or faster files. The results are (hopefully) functionally equivalent to the program without optimization. In practice optimizer bugs do occur. Recognizing things that can be optimized is actually rather difficult and is not always done correctly. Because of this, optimizing should be the last thing done to a program. Once a program works without optimization, then the optimizer can be used to try to improve the program. If the program doesn't work, then the optimizer can be blamed.

(It's a computer science joke that compilers also contain a *pessimizer* that breaks your code and introduces bugs. Unfortunately, nobody has been able to find the flag that turns this feature off.)

- g Turns on code generating options for debugging. Detailed information is stored in the object files and executable files about which lines in the source code file are associated with the machine code instructions. This can make programs comparatively large and slow. However, this information is used by a debugger to allow stepping through a program line by line as it executes, which is very useful. The `-g` option is incompatible with the `-O` option. They should not be used together.
- Wall  
Turn on all the warning messages possible. Most useful during the compilation phase, but also works during the linking phase.
- I directory\_name  
Use the given directory as a place to search for include files. This directory is used in addition to the standard system include directories. Multiple include directories are specified by using a separate `-I` option for each directory. There is no space between the `-I` option and the directory name.

The standard directories used by a compiler are usually built in to the compiler and may vary from machine to machine, but `/usr/include` is often one of the standard directories.

This option is useful only during the compilation phase.

#### **-llibrary\_name**

Link the given library into the program. A library is a collection of pre-compiled object code. This option is used only during the linking phase. This option comes at the end of the command line. Multiple libraries are included by using a separate `-l` option for each library.

The standard system libraries are usually found in `/lib` and `/usr/lib`. The names of the libraries take the form of `libname.a` or `libname.so`. The part of the library name after the `lib` and before the suffix is used as the name for linking. For example a common version of the math library is named `libm.a`. It is linked into a program using the `-lm` option.

#### **-Ldirectory\_name**

Use the given directory as a place to search for library files. This directory is used in addition to the standard system library directories. Multiple library directories are specified by using a separate `-L` option for each directory. There is no space between the `-L` option and the directory name.

This option is useful only during the linking phase.

## **Examples**

Here are some examples of using `g++` with many of the options given above.

For the compilation phase:

```
g++ -c -O -Wall -I my_include_dir myprog.C
```

compiles `myprog.C` to an object code file with optimization turned on, showing all warnings, and looking in the directory `my_include_dir` for any additional include files.

For the linking phase:

```
g++ -g -Wall -o fun fun.o support.o graphics.o -L/usr/lib/X11 -lX11 -lm
```

links the object code files `fun.o`, `support.o` and `graphics.o` into the executable file `fun`. All warnings are displayed and debugging information is preserved. In addition, the directory `/usr/lib/X11` is used to search for additional libraries and the libraries `X11` and `m` (`X11` graphics and math) are linked in as well.

[Web](#) [Images](#) [Groups](#) [News](#) [more »](#)[Advanced Search](#)  
[Preferences](#)

## Web

Results 11 - 20 of about 7,120 for "line by line" compiler. (0.20 seconds)

### TextFilt

... tool was designed to work on streams to support on-the-fly processing of **compiler** output ... After parsing the rulesets, the standard input is parsed **line-by-line**. ...

<http://textfilt.sf.net/> - 12k - [Cached](#) - [Similar pages](#)

### SIMULA Script Compiler

... code Provide a **line-by-line** dump of generated code. This option is primarily for debugging and maintenance purposes for the script **compiler**. ...

<http://home.comcast.net/~gep-2/scrcomp1.html> - 8k - [Cached](#) - [Similar pages](#)

### Peter's gdb Tutorial

... lines of code, so it gives the illusion of stepping through your code **line by line**. ... happen if you try to debug code that you turned **compiler** optimizations on ...

<http://www.dirac.org/linux/gdb/1-gdb.php> - 10k - [Cached](#) - [Similar pages](#)

### "Oz bugs" - compiler/343

... **Compiler** crash From: Leif Kornstaedt <kornstae@ps.uni-sb.de> Date: 11 May 1999 11:53:28 +0200 Reported by Thorsten Brunklaus: Feed the following **line by line** ...

<http://www.mozart-oz.org/cgi-bin/oz-bugs/compiler?id=343;user=guest> - 10k - [Cached](#) - [Similar pages](#)

### Conversion and Testing of F90 ARPS

... Read the code **line by line** and. ... Check with the F77 code when **compiler** Complains (When character variables with different lengths were declared in one single line ...

<http://www.caps.ou.edu/ARPS/F90ARPS.html> - 68k - [Cached](#) - [Similar pages](#)

### MatchActionProcessor (Jakarta ORO 2.0.6 API)

... The MatchActionProcessor class provides AWK-like **line by line** filtering of a text stream ... In fact, the default matcher and **compiler** used by the class are ...

<http://www.jakarta.org/oro/en/api/org/apache/oro/text/MatchActionProcessor.html> - 27k - [Cached](#) - [Similar pages](#)

### Delila Program: pbreak

... description The program pbreak will go through a file, **line by line**, looking for a ... The (@ form fools the **compiler**, and prevents it from thinking I'm doing ...

<http://www.lecb.ncicrf.gov/~toms/delila/pbreak.html> - 5k - [Cached](#) - [Similar pages](#)

### Fortran 77 Tutorial

... The **compiler** will try to detect if you access array elements that are out of ... You can step through a program **line by line** or define your own break points, you ...

[http://www.phy.nau.edu/~bowman/PHY520/F77tutor/20\\_debug.html](http://www.phy.nau.edu/~bowman/PHY520/F77tutor/20_debug.html) - 4k - [Cached](#) - [Similar pages](#)

### CodeGuru: C# and Intermediate Language

... **Line-by-Line** Analysis. Line Number, Analysis. ... The C# **compiler** won't compile whatever code is inside these comments. There are two kinds of comments in C#. ...

[http://www.codeguru.com/Csharp/Csharp/cs\\_syntax/anandctutorials/article.php/c5877/](http://www.codeguru.com/Csharp/Csharp/cs_syntax/anandctutorials/article.php/c5877/) - 42k - [Cached](#) - [Similar pages](#)

### COMIS Functions and the PIAF FORTRAN 77 Compiler

... **line-by-line** on each PIAF server when analysing an ntuple. Now it is possible to have this cut function compiled by the local FORTRAN 77 **compiler** on PIAF and ...

<http://www.wasd.web.cern.ch/wwwasd/paw/piaf/subsubsection3.4.2.5.html> - 4k - [Cached](#) - [Similar pages](#)

## Search for "line by line" compiler on [Images](#), [Groups](#), [News](#)




[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☐ The ACM Digital Library ☒ The Guide


**THE GUIDE TO COMPUTING LITERATURE**

[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

 Terms used **smart recompilation**

 Found **4,695** of **792,123**

Sort results by


[Save results to a Binder](#)
[Try an Advanced Search](#)

Display results


[Search Tips](#)
[Try this search in The Digital Library](#)
☐ Open results in a new window

Results 1 - 20 of 200

 Result page: [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) [9](#) [10](#) [next](#)

Best 200 shown

 Relevance scale ☐ ☐ ☐ ☐ ☐
**1 [Smart recompilation and the GNAT compiler](#)**

Franco Gasperoni, Patrick Bazire

 November 1994 **Proceedings of the conference on TRI-Ada '94**

 Full text available: [pdf\(722.53 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

The GNAT project at New York University is building a high-quality Ada 9X compiler, to be distributed free and with sources, following the successful mechanisms established by the Free Software Foundation for the GCC compiler. This paper describes the design of a smart recompilation system that is currently being implemented on top of the GNAT compiler. The foundations upon which smart GNAT rests are also the starting point for software engineering tools such as a ...

**2 [Smart recompilation: what is it?, its benefits for the user, and its implementation in the DEC Ada compilation system](#)**

Bevin R. Brett

 October 1993 **Proceedings of the conference on TRI-Ada '93**

 Full text available: [pdf\(811.08 KB\)](#) Additional Information: [full citation](#), [citations](#), [index terms](#)
**3 [Smart recompilation](#)**

Walter F. Tichy

 June 1986 **ACM Transactions on Programming Languages and Systems (TOPLAS), Volume 8 Issue 3**

 Full text available: [pdf\(1.56 MB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)

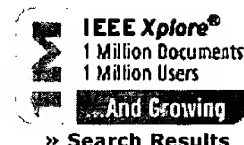
With current compiler technology, changing a single line in a large software system may trigger massive recompilations. If the change occurs in a file with shared declarations, all compilation units depending upon that file must be recompiled to assure consistency. However, many of those recompilations may be redundant, because the change may affect only a small fraction of the overall system. Smart recompilation is a method for reducing the set of modules that must be recompiled ...

**4 [Smart recompilation](#)**

Walter F. Tichy, Mark C. Baker

 January 1985 **Proceedings of the 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages**

 Full text available: [pdf\(835.58 KB\)](#) Additional Information: [full citation](#), [abstract](#), [citations](#), [index terms](#)

[IEEE HOME](#) | [SEARCH IEEE](#) | [SHOP](#) | [WEB ACCOUNT](#) | [CONTACT IEEE](#)[Membership](#) | [Publications/Services](#) | [Standards](#) | [Conferences](#) | [Careers/Jobs](#)**IEEE Xplore®**  
RELEASE 1.6Welcome  
United States Patent and Trademark Office[Help](#) | [FAQ](#) | [Terms](#) | [IEEE Peer Review](#)[Quick Links](#)

Welcome to IEEE Xplore®

- ☐ Home
- ☐ What Can I Access?
- ☐ Log-out

Tables of Contents

- ☐ Journals & Magazines
- ☐ Conference Proceedings
- ☐ Standards

Search

- ☐ By Author
- ☐ Basic
- ☐ Advanced

Member Services

- ☐ Join IEEE
- ☐ Establish IEEE Web Account
- ☐ Access the IEEE Member Digital Library

Your search matched **1** of **1011253** documents.A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance** in **Descending** order.**Refine This Search:**

You may refine your search by editing the current search expression or entering a new one in the text box.

☐ Check to search within this result set**Results Key:****JNL** = Journal or Magazine   **CNF** = Conference   **STD** = Standard**1 The binary compatibility standard***Anderson, A.; Cruess, M.; Wiencek, E.;*

COMPCON Spring '89. Thirty-Fourth IEEE Computer Society International

Conference: Intellectual Leverage, Digest of Papers. , 27 Feb.-3 March 1989

Pages:32 - 37

[\[Abstract\]](#)   [\[PDF Full-Text \(464 KB\)\]](#)   **IEEE CNF**[Home](#) | [Log-out](#) | [Journals](#) | [Conference Proceedings](#) | [Standards](#) | [Search by Author](#) | [Basic Search](#) | [Advanced Search](#) | [Join IEEE](#) | [Web Account](#) | [New this week](#) | [OPAC](#) | [Linking Information](#) | [Your Feedback](#) | [Technical Support](#) | [Email Alerting](#) | [No Robots Please](#) | [Release Notes](#) | [IEEE Online Publications](#) | [Help](#) | [FAQ](#) | [Terms](#) | [Back to Top](#)

Copyright © 2004 IEEE — All rights reserved

[Home](#) [Index](#) [Resources](#) [Contact Us](#) [Internet Search](#)**Scientific and Technical Information Center****[Patent Intranet](#) > [NPL Virtual Library](#) > [Request a Prior Art Search](#)****[Patents Home](#) | [Site Feedback](#)**[NPL Home](#) | [STIC Catalog](#) | [Site Guide](#) | [EIC](#) | [Automation Training/ITRPs](#) | [Contact Us](#) | [STIC Staff](#) | [FAQ](#) | [Firewall Authentication](#)**Request a Prior Art Search**

Search requests relating to **published applications, patent families, and litigation** may be submitted by filling out this form and clicking on "Send."

For all other search requests, fill out the form, print, and submit the printout with any attachments to the STIC facility serving your Technology Center.

**Tech Center:**

- ☐ TC 1600    ☐ TC 1700    ☒ TC 2100    ☐ TC 2600  
☐ TC 2800    ☐ TC 3600    ☐ TC 3700    ☐ Other

**Enter your Contact Information below:**Name: Employee Number: Phone: Art Unit or Office: Building & Room Number: **Enter the case serial number (Required):** 

If not related to a patent application, please enter NA here.

**Class / Subclass(es)** **Earliest Priority Filing Date:** **Format preferred for results:**☐ Paper    ☐ Diskette    ☒ E-mail**Provide detailed information on your search topic:**

- In your own words, describe in detail the concepts or subjects you want us to search.
- Include synonyms, keywords, and acronyms. Define terms that have special meanings.
- **\*For Chemical Structure Searches Only\***  
Include the elected species or structures, keywords, synonyms, acronyms, and registry numbers
- **\*For Sequence Searches Only\***  
Include all pertinent information (parent, child, divisional, or issued patent numbers) along with the appropriate serial number.
- **\*For Foreign Patent Family Searches Only\***  
Include the country name and patent number.
- Provide examples or give us relevant citations, authors, etc., if known.

- FAX or send the **abstract, pertinent claims** (not all of the claims), **drawings, or chemical structures** to your EIC or branch library.

**Enter your Search Topic Information below:**

When computer program source code is compiled, the compiler translates the source from a human readable format to a machine readable format called object code which is stored in a separate file. Object code is often divided into sections containing the actual machine program instructions (text), as well as variable information (data). Once a program is compiled into object code, it is difficult or impossible to reconstruct the original source file. The invention embeds the source code directly in the object code file as a separate section. When the original source file is further edited, the compiler compares the edited source file with the original source file stored in the object code, and only compiles those portions that are changed. The new compiled code section then replaces the old in the object file, and the source section of the object file is also updated to reflect the change. This is an effort to reduce compilation time. The key feature is that the source code is being stored in the object file.

Terms: compiler, object file format, object code, source code,

**Special Instructions and Other Comments:**

(For fastest service, let us know the best times to contact you, in case the searcher needs further clarification on your search.)

I'm usually in the office M-F 6:30-3

**Press ALT + F, then P to print this screen for your own information.**

**SEND****RESET**

USPTO [Intranet Home](#) | [Index](#) | [Resources](#) | [Contacts](#) | [Internet](#) | [Search](#) | [Web Services](#)

Last Modified: 12/05/2003 15:08:46